Received 21 July 2023, accepted 20 August 2023, date of publication 23 August 2023, date of current version 7 September 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3308036

RESEARCH ARTICLE

A Novel Deep Hierarchical Machine Learning Approach for Identification of Known and Unknown Multiple Security Attacks in a D2D Communications Network

S. V. JANSI RANI[1], IACOVOS I. IOANNOU[2,3], PRABAGARANE NAGARADJANE[4], CHRISTOPHOROS CHRISTOPHOROU[2,3], VASOS VASSILIOU[2,3], (Senior Member, IEEE), HARSHITAA YARRAMSETTI[1], SAI SHRIDHAR[1], L. MUKUND BALAJI[1], AND ANDREAS PITSILLIDES[2,5], (Life Senior Member, IEEE)

[1]Department of Computer Science and Engineering, Sri Sivasubramaniya Nadar College of Engineering, Chennai 603110, India
[2]Department of Computer Science, University of Cyprus, Nicosia 1678, Cyprus
[3]CYENS Centre of Excellence, Nicosia 1016, Cyprus
[4]Department of ECE, Sri Sivasubramaniya Nadar College of Engineering, Chennai 603110, India
[5]Department of Electrical and Electronic Engineering Science, University of Johannesburg, Johannesburg 2092, South Africa

Corresponding author: Iacovos I. Ioannou (ioannou.iakovos@ucy.ac.cy)

This work was supported in part by the European Union's Horizon 2020 Research and Innovation Program under Agreement N°739578; in part by the Government of the Republic of Cyprus through the Directorate General for European Programs, Coordination, and Development; in part by the European Union under Grant 101087257 and Grant 101095363.

ABSTRACT Intrusion Detection Systems (IDSs) have played a crucial role in identifying cyber threats for a very long time. Still, their significance has increased significantly with the advent of 5G/6G technologies, particularly Device-to-Device (D2D) communication. Multiple cyberattacks, such as Man in the Middle (MITM) attacks, Structured Query Language (SQL) injection attacks, Dictionary attacks, Distributed Denial of Service (DDoS) attacks, and others by using specific attack tools such as HULK, RUDY, and GoldenEye, that can cause rapid battery drain, rendering D2D network devices more prone to hardware failure or even to the dissolution of the D2D communication network affecting the operation and the performance of the mobile network. Using a Deep Hierarchical Machine Learning Model/Deep Hierarchical Neural Network (DHMLM/DHNN) technique, we develop an Intrusion Detection System (IDS) for D2D communication that, due to its hierarchical structure, is distinct from other comparable approaches. (i.e., Recurrent Neural Networks (RNN), Deep Neural Networks (DNN), Long short-term memory (LSTM)), has several advantages, including i) reduced training time (training time can be reduced by 56%.); ii) the ability to identify multiple types of attacks; iii) the ability to identify Zero-day/Unknown attacks (i.e., attacks that it has not seen before); iv) a more straightforward model design due to the low number of connections and neurons compared to other approaches (excluding RNN and LSTM), and; v) overall outstanding performance in terms of accuracy (i.e., 99.07%). The custom/unified data set used to train and evaluate the model was partially manually emulated and partially sampled from a large set (>95%) from the commonly used CIC-DDoS-2019 data set. The after-comparison final proposed model's 99.07% accuracy on this unified data set demonstrates the efficacy of our method. The model was also tested and demonstrated an astounding 99.63% accuracy for zero-day/unknown attacks.

INDEX TERMS 5G, D2D, D2D security, intrusion detection systems, multiple cyber attacks, hierarchical machine learning.

The associate editor coordinating the review of this manuscript and approving it for publication was Bilal Khawaja.

I. INTRODUCTION

Device to Device (D2D) networks [1], [2] represents a new communication paradigm for 5G and beyond cellular networks, as it enables devices near in proximity to connect via a direct link, as opposed to sending network packets through a Base Station (BS) or a core network (which causes considerable packet transmission delays), but the devices in order to construct a network that can access the internet they must have at least one connection towards the BS (one UE that can share its connection towards BS and act as D2D Relay or D2D MultiHop Relay as mentioned in [2]) and let the BS act as gateway. Moreover, D2D networks aim to enable direct device-to-device communication and reduce reliance on base stations; in certain scenarios, the presence of a base station may still be necessary to facilitate network management and coordination. In this manner, the geographical proximity between network nodes increases network coverage and transmission speed. However, the lack of a monitoring authority, the dynamic nature of D2D networks (in which devices can enter and exit the network at will), and the low processing capability of pervasive devices, i.e., User Equipment (UE), make them vulnerable to cyber-attacks and makes D2D network recovery difficult [3]. These attacks are diverse and include crippling large-scale operations such as DDoS attacks that can affect the operation of tech titans like Twitter, Amazon, and Cloud Flare. The alleged Google Attack on October 16, 2020, has unquestionably cemented DDoS attack detection as the top cyber security objective. Functionally, two primary types of IDSs are employed to detect cyber attacks. These are [4]:

- Anomaly Detection: In anomaly detection (unusual event detection), the IDS learns the signature of regular traffic and records departures from that signature as highlighted anomalies for further investigation to discover attack signatures.
- Misuse Detection: In misuse detection, the IDS learns the attack signatures and classifies attacks as they are detected to neutralise them with minimal impact on the network infrastructure.

Note that due to the nature of the architecture utilised to establish D2D communication networks, cyber-attacks, especially DDoS attacks, are worsened, making standardisation of security measures and secure data transmission between devices challenging. Several Machine Learning (ML) methods, such as Support Vector Machine, k-nearest neighbour, and Naive Bayes [5], have been utilised in D2D networks to accurately identify and classify network data, as well as detect assaults. Numerous algorithms adopt a shallow learning strategy, emphasising feature selection and feature engineering to produce optimal outcomes. However, shallow models cannot categorise vast quantities of live traffic in a real-time application environment. Deep learners, such as Neural Networks (NN), outperform shallow learners in extracting trends from enormous amounts of data to generate more realistic models. NNs can learn from training data given to them without requiring any prior knowledge.

Hierarchical decomposition through Hierarchical Machine Model simplifies complex issues by breaking them into smaller, interconnected ones. The smaller problems are solved independently, and the answers are recombined to solve the original problem. Consequently, one of the primary objectives of hierarchical learning is the reduction of computing complexity. Using a collection of multi-level classifiers can lower the cost of learning, according to the proposed approach. Undoubtedly, the collection of multi-level classifiers constitutes a hierarchical learning structure. Experimental findings in handwritten Chinese character recognition and image retrieval are provided to validate the approach's utility [6], [7]. Hierarchical learning, inspired by human learning, is one of the techniques for enhancing machine learning performance [8], [9], [10], [11], [12].

The novelty of the investigation is that with the use of a DHMLM/DHNN (called also DHNN) approach, we develop an intrusion detection system (IDS) for D2D communication that, due to its hierarchical structure, has several advantages over other comparable approaches, such as reduced training time, the ability to identify numerous types of assaults on security, the ability to identify Zero-day attacks (i.e., attacks that it has never seen before), a more straightforward model design due to the low number of connections and neurons and remarkable overall accuracy. The key contributions of this paper are summarised below:

- Introduced (to the best of our knowledge) the DHMLM in various attack identification in D2D communication networks.
- Using the DHMLM model, we develop an IDS for D2D Networks that, due to its hierarchical structure, in contrast to other related approaches, has various advantages, including:
  - Reduced training time. More specifically, we achieved more significant computation optimisation with the use of a hierarchical structure model that permits greater flexibility and customisation (as demonstrated in Section V-D) rather than using a shallow DNN [13] or any other approach.
  - Ability to identify multiple types of Linux tool-based security attacks and with greater accuracy (i.e., 99.07%; see Section VI-B) than the DNN, RNN, and LSTM (see Section VI-B2) and the rest of the other investigated approaches (see Section VI-B). More specifically, the attacks identified are i) Man in the Middle attack; ii) SQL injection attack; iii) Dictionary attack; iv) various types of DDoS attacks such as UDP, SSDP, Synchronisation (SYN), NetBIOS, Portmap, Trivial File Transfer Protocol (TFTP), LDAP, and MSSQL; and v) various Kali as HULK, RUDY, and GoldenEye.
  - Ability to recognise and evaluate unknown threats as attacks (see Section VI-B5) [14].
  - Provides a simpler model design due to the low number of connections and neurons used.
- We have created a novel data set by combining the most common attacks of the CIC-DDoS-2019 data set [15]

and emulated attacks with the help of common Kali Linux tools and a D2D Client, thereby making the model immune to data distribution variability which leads to fewer false alarms and better accuracy.

The rest of the paper is structured as follows: Section II provides some background information related to D2D communication networks, Intrusion Detection Systems, the investigated approaches used in this research, and the examined cyber-attacks and other types of attacks explored in the paper (see Section II-A4). Additionally, Section II shows the related work in the open literature related to IDS and a comparison table of related works that utilised ML used at IDS along with our investigation approach (DHMLM) of this paper. The problem description, primary objective, and assumptions used in our investigation are described in Section III. The end-to-end workflow of our system is described in Section IV. The methodology of the approach, including the hierarchical architecture of the model, the emulation setup, the unified data set generation process used, the feature selection, and the DHMLM's, RNN, and LSTM workflow/architectural structure, are elaborated in Section V. The efficiency of the investigated approaches is examined, evaluated, and compared in Section VI. Finally, Section VII includes concluding remarks and our future directions.

II. BACKGROUND KNOWLEDGE AND RELATED WORK

This section provides the context for the primary concepts discussed in the paper and used in our solution approaches as background work. It also discusses recent results from relevant, high-quality papers on attack identification and intrusion detection in D2D and non-D2D communication networks.

A. BACKGROUND KNOWLEDGE

In this section, we provide some details about D2D, IDS, and the insides of our investigated machine learning techniques (i.e., RNN, LSTM, and deep hierarchical machine learning). Moreover, we provide some background information about the investigated Cyber-Attacks.

1) D2D COMMUNICATION NETWORKS

D2D networks are transparent to cellular networks (such as Base Stations) and operate independently in licensed (inband D2D) and unlicensed (outband D2D) spectra, and allow proximate devices to bypass BS and establish direct links between them. Also, D2D communications have the possibility of any device being directly connected to BS and becoming a gateway to others (share their connection, act as relay stations, or directly communicate and exchange information). D2D communication has applications in pervasive social networking, emergency rescue, location-based services, and LTE-Advanced networks' underlay networks. The standardisation of D2D is in progress. Due to automatic security and reduced computational power, D2D links are more vulnerable. D2D networks support three application scenarios: in-coverage, relay-coverage, and out-of-coverage. In in-coverage, UEs are within the cellular network's coverage, link establishment is operator (BS) controlled, and the band used is licensed cellular spectrum. D2D communication is also useful for reducing traffic by rerouting it. Relay coverage is the same, but when UEs are at the network's edge, the connection is poor, and D2D improves it. There is no UE network connectivity outside of coverage. Additionally, D2D communication is useful when natural disasters destroy the underlying infrastructure because it operates autonomously and does not require BS intervention. It also employs a dedicated band.

2) INTRUSION DETECTION SYSTEMS

Intrusion Detection systems are utilised to monitor and detect network traffic to identify malicious users through anomaly detection and misuse detection. In misuse detection, predefined models are compared to current user activity obtained by continuously monitoring network data. IDS are used to detect common, well-known intrusions. Anomaly detection detects novel intrusions by comparing normal traffic patterns with current traffic patterns and utilising a threshold to determine if the deviation is within the allowed threshold. Ideally, a real-time IDS should possess both of these features.

3) INVESTIGATION APPROACHES BASED ON THE MACHINE LEARNING

This section provides background information on the ML NN (i.e., RNN) and ML DNN (i.e., LSTM, DHMLM, and DNN) approaches we use to tackle the problem of detecting multiple security attacks in D2D communications.

a: RNN

RNN [16], [17] is a type of artificial neural network in which the connections between nodes can create a cycle, permitting the output of some nodes to affect subsequent input to the same nodes. This enables the entity's dynamic temporal behaviour. RNNs are dynamic systems where the internal state varies at each classification time step due to the circular connections and potential self-feedback connections between neurons in the upper and lower layers. These feedback connections allow RNNs to convey data from past occurrences to processing stages that are now occurring. Therefore, RNNs construct a memory of time series events. If a Recurrent neural network has time delays or feedback loops, a different network or graph may be substituted for the store. Note that LSTM networks with recurrent gated units contain regulated states known as gated states or gated memory. Using their internal state (memory), RNNs constructed from feedforward neural networks can manage input sequences of varying lengths. As stated previously, recurrent neural networks have an infinite impulse response, while convolutional neural networks have a finite impulse response; however, both network types exhibit dynamic temporal behaviour. In addition, RNNs can be serially interconnected; therefore, in the case of two RNNs with a single layer, the remaining RNNs range from partially linked to wholly connected. Comparable to a three-layered neural network, the Elman network[1] retains the hidden layer outputs within context cells and returns the context cell output and the signal's provenance to the hidden neuron. Each neuron in the hidden layer receives input from both neurons in the context layer and the input layer. Also, it adds a connecting layer to the hidden layer of the feedforward network as a time delay operator to memorise. As a result, it makes the system exhibit time-varying characteristics and stronger global stability. Elman networks can be trained using standard error backpropagation, with the output of the context cell considered an additional input.

b: LSTM

Long short-term memory (LSTM) is a feedback-connected artificial neural network. Over a thousand distinct time steps, LSTM can learn to bridge tiny temporal delays. The LSTM architecture is intended to equip RNNs with a short-term memory that can last for thousands of timesteps, hence the name "long short-term memory. A typical LSTM unit comprises a cell, an input gate, an output gate, and a forget gate. The network's connection weights and biases change once per training session, similar to how physiological changes in synaptic strengths store long-term memories; the network's activation patterns change once per time step, similar to how minute-to-minute changes in the brain's electrical firing patterns store short-term memories. Long-term, short-term memory is a gradient-based technique that avoids the vanishing gradient problem produced by the derivative of the activation function used to construct a neural network and, ultimately, the LSTM [19].

The system uses Constant Error Carousels (CECs) to ensure a consistent flow of errors within particular cells. The cell state is defined by activating a CEC by the input gate. Multiple gate devices regulate cell access by determining when admittance is permitted. This is the most important feature of LSTM, which enables long-term storage of short-term memory. As we still manage the connections between other units and a unit named, for example, u, the various LSTM network components are utilised here. By extending the CEC with input and output gates connected to the network's input layer and other memory cells, LSTM can avoid the issue of conflicting weight updates. This results in the creation of a memory block, which is a complex LSTM unit. The input gates, which are fundamental sigmoid threshold units with an activation function range of [0, 1], scale network signals for the memory cell; activation is near zero when the gate is closed. In addition, they can learn how to avoid extraneous signals from interfering with the stored data in the predefined unit u. The cell remembers values across arbitrary time intervals, and the three gates control data flow into and out of the cell. The output gates can learn how to buffer neighbouring memory cells from u-caused disruptions by regulating access to the memory cell's contents. The major role of multiplicative gate units is to authorise or refuse access to the CEC's continuous error flow [19].

[1]Elman neural network [18] is a feedback neural network that is optimised based on the research of backpropagation (BP) neural network by Elman in 1990.

c: DEEP HIERARCHICAL MACHINE LEARNING MODEL/DEEP HIERARCHICAL NEURAL NETWORK

This section provides information about Deep hierarchical Machine Learning. A brief description of Artificial Neural Networks (ANNs) along with the traditional DNNs should be provided to fully understand the concept because these are our approach's building blocks.

ANNs are supervised machine learning models inspired by our brain's biological network of neurons. They consist of a layer that receives external data (i.e., the input layer), a layer that produces the final result (i.e., the output layer), and zero or few hidden layers in between them. Each layer is composed of nodes or neurons that are connected according to a predetermined weight and threshold. If the output of a particular node exceeds the threshold, the node is activated. Because ANNs are supervised learning models requiring substantial training data to improve their accuracy. A DNN is an ANN with more layers and is, therefore, capable of identifying more complex data patterns.

Traditional DNNs are frequently utilised in attack and threat detection [20]. The deep learning architecture uses many neurons to introduce non-linearity in classification hypotheses, thereby increasing classification complexity. It consists of one input layer and an arbitrary number of hidden layers containing neurons. This network has complete connectivity. Every connection has a corresponding parameter. The output layer may be a multi-class or binary classification that makes predictions using the parameters learned by all connections. However, weight or parameter sharing between the DNN nodes leads to a restricting interdependence between them or imposes constraints on their interdependence. Also, the complete network must be retrained whenever a change must be incorporated into a DNN. The network also exhibits a marked increase in complexity when more nodes and layers are added, increasing trainable parameters. In addition, the DNN's black box design prevents it from comprehending or identifying the hierarchical nature of the attacks.

In hierarchical machine learning-based models, supervised learning (classification) can be viewed as discovering a mapping from a low-level feature space to a high-level conceptual space or from an adequate space to its quotient space (a coarse space). There is a large semantic gap between the low-level feature spaces and the conceptual spaces, making it difficult and inefficient to locate the mapping. Human learning features are applied to lower the computational complexity of machine learning. In human learning, people always employ a multi-level learning method, consisting of multi-level classifiers and multi-level features, instead of a single level, i.e. learning at distinct grain-size spaces. This type of machine learning is known as hierarchical learning.

Therefore, hierarchical learning is an effective method for enhancing machine learning.

A DHMLM is a machine learning model with a hierarchical data-driven structure. In contrast to traditional DNNs, which have a black box model, a-priori data knowledge is used to structure the networks, thereby enhancing our understanding of the model's operation and efficacy. A Deep Hierarchical ML model is a classification model that exploits hierarchical patterns within the data. DHMLM is comprised of learners of multiple levels. The models in question could be homogeneous or heterogeneous and consist of any machine learning algorithm. In this paper, we leverage the inherently hierarchical nature of the data and use a DHMLM-based approach that achieves modularity by splitting the realisation of the IDS task into three DNNs components to implement an efficient and real-time IDS. The reasons for choosing DHMLM to design our approach are the following:

- The DHMLM is capable of identifying unknown attacks without the need for prior training (see Section VI-B5).
- The DHMLM is hierarchical and consists of separate levels of DNNs. The model's retraining consumes significantly less time as we are only required to train one or two relatively shallow DNNs. Furthermore, the overall training time of the DHMLM is almost halved compared to a similar traditional DNN.

4) INVESTIGATED CYBER ATTACKS

A cyber-attack is any malicious attack aimed at disrupting operations, gaining access to data, or damaging information. Cybersecurity threats are the greatest obstacle for any company operating in the Information Technology (IT) sector. Globally, it is estimated that the total damages caused by cyber events amount to approximately \$8.5 billion [21]. The most common cyber attacks are MITM, DoS, DDoS attacks, SQL injection attacks, Password attacks, Phishing, Malware, and Ransomware attacks.

In this paper, we have examined and compiled a data set containing MITM attack using Ethernet Capture (ETTERCAP) tool, HTTP DDoS attack using R U Dead yet (R.U.D.Y.) tool, DDoS attack using GoldenEye tool, Hypertext Transfer Protocol (HTTP) DDoS attack using Flood, DoS HTTP attack using HTTP-Unbearable-Load-King (HULK) DDoS tool attack, TFTP amplification DDoS attack using Low Orbit Ion Cannon (LOIC) tool, UDP (User Datagram Protocol) Flood DoS attack using Hping3 tool, Lightweight Directory Access Protocol (LDAP) reflection and amplification DDoS attack using Distributed Reflection Denial of Service (DrDoS) tool (called r2dr2-udp-drdos-tool), Simple Service Discovery Protocol (SSDP) distributed reflective DDoS attack using Saddam tool, Network Basic Input/Output System (NetBIOS) DDoS attack using XOIC tool, Port Mapper (Portmap) DDoS Attack attack using Rpcinfo tool, Synchronisation (SYN) flood DDoS attack using Hping3, Structured Query Language (SQL) injection attack using #RefRef tool, Microsoft SQL Server Resolution Protocol (MC-SQLR) amplification for ms-sql server DDoS attack using mssqldos tool and Password attack which is a type of dictionary attack using xHydra tool, which we briefly describe below. Also, some terms considered in our investigation are defined. The list of attacks with their description and the related protocols considered in our research, along with the related tools used in the emulation, are described below:

- **Man-in-the-middle (MITM) attack [22]:** A Man-in-the-middle attack is an attack in which an attacker eavesdrops on a conversation over the network between a computer with another computer using a protocol (such as address resolution protocol) or a user and a web application. The attacker secretly collects data such as personal information and passwords by sifting through the network packets exchanged between the application and the user. The adversary can also use this information to assume the user's identity and conduct illegal operations [22]. The related protocol is Address Resolution Protocol(ARP). The tool that is used in the emulation for doing MiTM is the Ethernet Capture (ETTERCAP). ETTERCAP is a powerful network interception tool used for network monitoring, analysis, and penetration testing. It stands for Ethernet Capture and is primarily designed to perform man-in-the-middle (MITM) attacks on computer networks. ETTERCAP allows security professionals to intercept, sniff, and manipulate network traffic flowing between different devices on a network. With ETTERCAP, you can analyse and capture various network protocols such as TCP/IP, Address Resolution Protocol (ARP), Internet Control Message Protocol (ICMP), Dynamic Host Configuration Protocol (DHCP), and others. It operates by placing the attacker's machine in the middle of a communication flow between two devices, allowing it to intercept and modify network packets in real-time. Another tool that can be used in the MiTM attachment is Hyenae. Hyenae is a platform-independent, extremely flexible network packet generator. It enables the reproduction of multiple MITM, DoS, and DDoS attack scenarios, including a clusterable remote daemon and an interactive attack assistant.
- **SQL Injection attack [23]:** An SQL injection attack exploits vulnerabilities in a SQL database server and enables the attacker to inject statements that allow him to insert large amounts of fake data into the database or extract and delete data from the database, thereby compromising the data's integrity [24]. The related protocol is Hypertext Transfer Protocol (HTTP). The tool that is used in the emulation for SQL injection is the #RefRef. #RefRef is a Perl-based DoS attack tool created by the Hacktivist group 'Anonymous' that exploits a MySQL vulnerability and it leverages it to execute a SQL injection employing the MySQL BENCHMARK() function, allowing for repeated execution of an expression to cripple the website through MySQL. #RefRef exploits the BENCHMARK() function, which enables the repetitive

execution of an expression to exhaust the resources of a targeted server. If the server's infrastructure employs MySQL and is vulnerable, a significant disruption can be caused by a few machines. A tool that can be used in the SQL injection attack, also, is the sqlmap. It detects and exploits SQL injection vulnerabilities and takes over database servers automatically. It includes a powerful detection engine and a vast array of switches for database fingerprinting, data retrieval, accessing the underlying file system, and executing commands on the operating system via out-of-band connections.

- **Password attack [25]:** As the name suggests, password attacks are brute force attacks used to steal a victim's passwords. It employs a trial-and-error algorithm for guessing passwords and login credentials [26]. Password attack is a type of dictionary attack. The tool that is used in the emulation for making Password Attacks is xHydra. The xHydra is a powerful graphical user interface (GUI) application with Hydra tool as the back-end for brute force attacks and password decryption. Hydra is used for online or offline password attacks. The xHydra tool is used to commence brute-force password attacks. Additionally, xHydra is primarily used to evaluate the efficacy of passwords and authentication systems through automated and systematic attempts to determine login credentials. The xHydra is based on the well-known and widely used Hydra application, which is a command-line password decryption utility. However, xHydra provides a user-friendly interface that facilitates the configuration and execution of brute-force attacks. The application supports multiple attack protocols, including SSH, FTP, Telnet, and HTTP. It permits users to specify a list of target IP addresses or hostnames, username lists, and diverse password generation techniques, such as dictionaries, masks, and combination attacks. A tool that can be used in the password attack, also, is the Medusa. Medusa is a rapid, modular, and parallel brute-force password encoder. The Medusa utility is a lightweight and extremely effective instrument and is used to brute-force credentials across as many protocols as possible, resulting in the remote execution of code.
- **DoS attack [27]:** A DoS is a malicious and targeted attack that bombards a network with repeated synthetic requests to disrupt or even shut down the network. This prevents network users from making use of network resources. It does not result in data theft or loss, but the cost to restore networks and services is significant. The related DoS tools, the associated attack, and the associated protocols (in the parenthesis) are:

  - **Http-Unbearable-Load-King (H.U.L.K) tool, HTTP attack (HTTP protocol) [28]:** HULK is a Denial of Service (DoS) tool used to attack web servers by generating unique and obfuscated traffic volumes. Thus, it uses some obfuscation techniques to mask the Hulk server generating the requests meant to overwhelm the web server hosted on the victim device. The URL in the request is also modified to make the URL look legitimate enough to be processed.
  - **Hping3 tool, UDP (User Datagram Protocol) Flood attack (UDP protocol) [29]:** Hping3 is a network tool able to send custom ICMP/UDP/TCP packets and to display target replies like ping does with ICMP replies. A UDP flood attack is a type of DoS attack in which many User Datagram Protocol (UDP) packets are sent to a targeted server to overwhelm that device's ability to process and respond.
  - **R U Dead Yet (R.U.D.Y.) tool, HTTP attack (HTTP protocol) [30]:** By submitting lengthy form fields, R.U.D.Y. is designed to cause a web server to collapse. R.U.D.Y. is a slow-rate DoS attack that opens relatively few connections to the targeted server or website over a period of time and leaves the sessions open as long as possible. The number and length of open HTTP sessions exhaust the target's resources making it unavailable for legitimate traffic.

- **DDoS attack [31]:** The goal of a DDoS attack is to disrupt a network by employing multiple systems to perform various DoS attacks on the same network [32]. Distributed Denial of Service Attack is an attack that aims to disrupt the regular operation of a web server by consuming its resources and violating the Availability security principle by preventing legitimate users from accessing the web server's services. DDoS has the following types: i) Volume-based attacks, ii) Application layer attacks, and iii) Protocol attacks. The related DDoS tools, the associated attack, and the associated protocols (in the parenthesis) are:

  - **Low Orbit Ion Cannon tool, TFTP amplification attack (TFTP protocol) [33]:** Low Orbit Ion Cannon (LOIC) tool is used for network stress testing, and capable of initiating TCP, UDP, and HTTP GET floods. A DDoS attack that engages TFTP servers (that use UDP) connected to the Internet. It makes a default request for a file, and the victim TFTP server returns data to the requesting target host as a result of this request, regardless of a file name mismatch, resulting in a waste of time.
  - **GoldenEye Tool, DDoS attack (HTTP protocol) [34]:** Goldeneye is a tool that incapacitates a web server by transmitting legitimate HTTP traffic. It is an open-source tool written in.NET Core that allows users to create legitimate HTTP requests using a botnet. It requires only a few hundred requests at regular intervals after establishing a complete TCP connection. Consequently, the application does not need to utilise a substantial amount of traffic to exhaust the available server connec-

tions. A DDoS attack tool that uses legitimate HTTP traffic to take down a web server. It sends requests to the target and generates heavy traffic using botnets.

- **FLOOD Tool, DDoS attack (HTTP protocol on GET and POST requests) [34]:** FLOOD is a tool designed specifically to launch HTTP deluge/flood attacks. It is a Python script-based tool with HTTP GET and POST types of requests. The HTTP Flood attack uses the HTTP GET request on the FLOOD tool. HTTP deluge assaults are denial-of-service (DoS) attacks in which many ostensibly legitimate HTTP queries are sent simultaneously or in rapid succession to a target web server, overwhelming its resources and causing it to become unresponsive or collapse. An HTTP deluge attack aims to disrupt the accessibility and functionality of a targeted web application or service, thereby denying access to authorised users. These attacks frequently exploit vulnerabilities in the server's management of HTTP requests, such as processing or resource allocation inefficiencies.
- **Saddam tool, SSDP) distributed reflective[2] attack (Universal Plug and Play (UPnP) protocol) [35]:** Saddam is a tool that uses UDP reflection and amplification in Domain Name Server (DNS), Network Time Protocol (NTP), Simple Network Management Protocol (SNMP), and Simple Service Discovery Protocol (SSDP) servers to saturate and overwhelm a target's internet uplink. A server capable of simulating UDP traffic, which entails transmitting Internet Protocol (IP) packets to the internet with another user's IP address, would be required to archive this. Specific DNS servers do not respond to DNS ANY queries. The second requirement is a list of servers that respond to UDP queries, presumably without rate limiting and with a high reflection factor. A DDoS attack exploits UPnP networking protocols to send an amplified amount of traffic to a targeted victim. It overwhelms its infrastructure because the attack attacks a protocol under/part of the Universal Plug and Play Protocols.
- **Distributed Reflection Denial of Service (DrDoS) tool (called r2dr2-udp-drdos-tool), LDAP reflection and amplification[3] through connectionless LDAP, or CLDAP attack (LDAP protocol) [36]:** DrDoS tool techniques typically involve multiple victim devices that unwittingly contribute to a DDoS attack against the perpetrator's intended target with increased assaults. Anonymity is a benefit of the DrDoS attack method. In a Distributed Denial of Service (DDoS) attack, the target website appears to be under attack by the victim servers rather than the attacker. So, this DDoS attack sends multiple requests to various servers on the Internet using a spoofed IP address of the victim. This causes the web servers to send their responses to the spoofed address, i.e. the victim instead of the attacker. Note that in this attack the Saddam-new tool can be used (as seen in the previous list item).
- **XOIC tool, Network Basic Input/Output System (NetBIOS) attack (NetBIOS protocol) [37]:** XOIC executes attacks against the target IP address based on the port and protocol selected by the user. The attack mode of the XOIC includes an Internet Control Message Protocol (ICMP) Flood. XOIC's Test Mode can be used to assess the effectiveness of the host launching the attack. In a NetBIOS-based DDoS attack, the attacker sends many queries to the port on which the NetBIOS service runs, increasing network traffic. The protocol that the attacker attacks is a type of User Datagram Protocol. NetBIOS is a service on Windows that allows file sharing through the Internet.
- **Hping3, Synchronisation (SYN) flood (Transmission Control Protocol (TCP) protocol) [38]:** Hping3 is a network utility that can send customised ICMP/UDP/TCP packets and display target replies as ping displays ICMP replies. It supports fragmentation, arbitrary packet size, and content and can transfer files using supported protocols. An SYN flood is a DDoS attack in which the attacker sends only initial connection (SYN) requests. The SYN is a Transmission Control Protocol message type. The victim responds to these requests and keeps all its ports open for a response making it unable to respond to legitimate traffic.
- **Mssqldos tool, Microsoft SQL Server Resolution Protocol (MC-SQLR) amplification for ms-sql server (MC-SQLR protocol) [39]:** mssqldos operates by broadcasting the CLNT_BCAST_EX packet, which requests a directory of database instances on the network and connection informa-

[2]An attacker conducts a reflection attack by impersonating a target's IP address and sending a request for information, typically via the User Datagram Protocol (UDP) or, in some instances, the Transmission Control Protocol (TCP). The server then transmits a response to the IP address of the target in response to the request. This "reflection" - employing the same protocol in both directions - is the reason why this is known as a reflection attack. Any server offering UDP or TCP-based services may be targeted as a reflector.

[3]Amplification attacks generate a large number of packets to overwhelm a website without alerting an intermediary. This occurs when a service transmits a lengthy response in response to an adversary's "trigger packet" request. An attacker can transmit tens of thousands of these queries to vulnerable services using readily available tools, resulting in responses that are considerably bigger than the original request and significantly increasing the target's size and bandwidth.

tion. Mssqldos requests the transmission of the phonebook to the target IP address thousands of times per second. The DDoS attack exploits the MC-SQLR, which responds with an exhaustive list of all database instances when a client requests data from a particular database. MC-SQLR is a protocol designed to use the User Datagram Protocol (UDP) to perform the attack. In this attack, hackers spoof their Internet Protocol address with the address of the target or victim and request from the MSSQL database server the list of online databases. This results in an amplified attack.

- **High Orbit Ion Canon (HOIC) tool, Hypertext Transfer Protocol (HTTP) Flood [40]:** HOIC is an update of the LOIC and was created to use accelerator scripts that allow attackers to distribute attack traffic and conceal their collocation, requiring only a single user's coordination. The HTTP flood is a type of Distributed Denial of Service (DDoS) attack in which the attacker exploits seemingly-legitimate HTTP GET or POST requests to attack a web server or application. The attacker often employs a botnet to send large volumes of these requests.
- **Rpcinfo tool, Port Mapper (Portmap) Attack (Remote Procedure Call (RPC) protocol) [41]:** RPCInfo employs Open Network Computing (ONC) Remote Procedure Call (RPC) protocols to communicate with the Portmapper daemon/service, which operates on port 111 by default on Unix and Linux machines. A Portmapper output is detailing all active and registered daemons. Includes RPC TCP Ping and UDP Ping, which query the operating port versions of daemons. The port mapper (rpc.portmap, portmap, or rpcbind) is an ONC RPC service that operates on network nodes that provide other ONC RPC services. Thus, Portmap is a mechanism by which Remote Procedure Call (RPC) services register to allow calls to be made to the Internet. In this DDoS attack, victims are sent large amounts of responses from Portmap servers using Transmission Control Protocol or User Datagram Protocol, making web-based services unusable.

B. RELATED WORK

This section summarises the learners/approaches found in the literature that utilise AI/ML (e.g., Deep NN, RNN, LSTM and Hierarchical) techniques along with their application in Intrusion Detection Systems for detecting DDoS, DoS, Password, MiTM, SQL Injection and other types of attacks. Additionally, security approaches for D2D networks have been considered in the investigation because our setup is focused on the D2D communication network.

The contribution of the paper [42] is the introduction of a structure-oriented network known as a Hierarchical Neural Network (HNN) that overcomes the shortcomings of the Traditional Neural Network (TNN). TNNs' lack of a consistent internal structure provides no information about the task. With the aid of apriori knowledge, these networks can be structured. Thus, the paper's contribution was to reduce the number of HNN nodes and connections, thereby reducing computational costs.

Two HNN models, Parallel Hierarchical IDS (PHIDS) and Serial Hierarchical IDS (SHIDS), are proposed by authors in [43]. SHIDS used an anomaly classifier to detect intrusion packets stored in a database and clustered using a clustering algorithm. When the number of attacks reaches a predetermined threshold, the networks retrain. PHIDS rectifies SHIDS's single-point failure problem by employing a three-level network structure, thereby significantly improving accuracy by reducing error accumulation. The detection rate for SHIDS and PHIDS was greater than 98%, and the rate of false positives was less than 10%.

Authors in [44] investigate the possibility of a Neural Network self-organising into a self-consistent, hierarchical internal structure. This experiment is conducted based on the Adaptive Resonance Theory (ART) Neural Network. The authors' model is able to generate new output nodes without losing track of old categories. Experiments conducted on the Zoo ML data set [45] produced a distinct hierarchy.

Authors in [46] introduce a feature selection methodology based on a wrapper method called Cuttlefish Algorithm (CFA) and use Feature Grouping based on Linear Correlation Coefficient (FGLCC) to reduce the complexity of the same. CFA is used to search for the optimal subset of features to improve the classifier's accuracy. On the KDD Cup'99 data set, the combined feature selection algorithm outperforms Feature Grouping based on Mutual Information (FGMI) and Label Construction for Feature Selection (LCFS). FGLCC-CFA had a detection rate above 95%, whereas LCFS and FGMI had detection rates below 90%.

In the experiment conducted by authors in [47], five Android devices are used to conduct DoS and DDoS attacks over a shared access point. It was discovered that the attacks were taxing the web servers hosted by the victim devices, resulting in their restart. Due to heating issues, they also caused the devices themselves to restart. This highlighted the negative impact of DDoS attacks on victimised devices.

The authors in [48] propose a new feature selection algorithm that calculates Attribute Ratio (AR) based on each class's average and frequency of attributes. The AR values are then ranked, and the most advantageous features are chosen. This technique was tested on Information Gain, Gain Ratio, and Correlation-based Feature Selection. Using the top 22 features and a J48 decision tree, implementing the iterative dichotomise (ID3) algorithm, 99.74% accuracy was achieved on the NSL-KDD data set, which was superior to the competition.

Authors in [49] apply multiple machine learning techniques to a newly generated data set titled Game Theory and Cyber Security (GTCS). The problem is subdivided into subproblems, and one ensemble module is assigned to each subproblem. For the above problem, the Naive Bayes, Logistic Multi-Layer Perceptron (MLP), k–nearest neighbour (IBK), Sequential Minimal Optimisation, and J48 are used as classifiers. The ensemble model utilised the top three performers, MLP, J48, and IBK, to enhance precision. The ensemble model outperformed all individual models.

Using Call Detail Records (CDR), the authors in [50] propose a novel Residual Convolutional Neural Network (CNN) with 50 layers for the detection of attacks such as Silent Call, Signaling, and SMS flooding against 4G infrastructure (CDR). In 2015, the experiments were conducted on data obtained from Telecom Italia. Due to the relatively small size of the input images, this paper also develops a deep rudimentary CNN (DRC) model. In a blended attack, the DRC achieved 91% accuracy while the more complex RNN achieved 97% accuracy.

Authors in [51] propose the multi-level intrusion detection system (ML-IDS), which employs two types of techniques: flow-based, which operates based on a set of rules, and protocol-based, which identifies illegal state transitions. ML-IDS employs decision correlation metrics to measure attributes that identify an ongoing attack. The proposed method uses multiple levels of data analysis to reduce the number of false alarms. The Payload Behaviour Analysis, Risk and Impact Analysis, and Adaptive Learning modules have not yet been implemented into the ML-IDS.

Authors in [52] introduce a multiple-kernel learning DDoS attack detection method that combines two multiple-kernel learning models with adaptive feature weights and an algorithm to extract five features. Multiple kernels are derived from the single kernel SVM model. In single-kernel SVM, samples are mapped to high-dimensional spaces using a single-kernel function. The SimpleMKL model is appropriate for all dimensions with a weight value of 1 but cannot exert distinct properties. Thus, two classifiers were required: one for interclass mean-squared difference growth (M-SMKL) and another for intra-class variance descent (S-SMKL). The data set used was the "DDoS Attack 2007" from Cooperative Association for Internet Data Analysis (CAIDA). Ensemble learning trains the two distinct classifiers, demonstrating their superior performance over SVM.

In [20], authors, claim in the paper, that by using DNN, a sort of deep learning model enables the development of a flexible and effective IDS for detecting and classifying unanticipated and unpredictable cyberattacks. However, they did not add simulations to prove their statement, and they do not clearly show the proof of concept.

Despite the vast amount of research in this field and the strengths of each of the methodologies discussed above, the examined models' computational complexity, training process, and the overall time needed for their training is rarely elaborated on. In our approach, we seek to shed light on and improve the area of model complexity and utilise an HML Model to identify unknown attacks and reduce the retraining time. Moreover, we have also focused on computation optimisation and outperforming a traditional DNN. Lastly, we have attempted to reduce false alarms by combining differently distributed data sets that will enable the model to make generalisations for more than just one distribution.

In the survey article [53], the authors present state-of-the-art solutions for optimising the orchestration in Federated Learning (FL) communications, focusing on deep reinforcement learning (DRL)-based autonomy approaches. The correlations between the DRL and Federated Learning (FL) mechanisms are described within the system architectures of selected literature approaches that have been optimised. To illustrate the applicability of DRL-assisted control to self-organising FL systems, observable states, configurable actions, and target rewards are investigated. Overall, the survey paper introduced some interesting concepts related to the IDS.

Note that this is precisely what our research has as a target, to act autonomously and distributed utilising the Deep Neural networks and especially the hierarchy to achieve known and unknown attack identification. In the future, our investigation has as its target to investigate the incrementally improve our model to not only identify the unclassified attacks but learn to classify them with the use of Federated Learning (FL). We will further discuss the statement in Section VI-B5.

1) EXISTING METHODS FOR IDENTIFYING NUMEROUS (ONE OR MORE) TYPES OF ATTACKS IN D2D ENVIRONMENTS

To the best of our knowledge, none of the investigations in the literature tackle multiple attacks, more than two using ML.

The authors of [47] analysed the impact of DoS attacks on a D2D underlying network. Their experimental results indicate that malicious individuals can force a mobile device to drop its Wi-Fi connection to an access point (AP) without being recognised by either the AP or the cellular network. They emphasise that these preliminary results will inspire further investigation in this burgeoning sector.

The primary objective of [54] was to expose security vulnerabilities, specifically DoS attacks, in device-to-device communication and to identify relevant security practices that can be used to protect this communication network by securing Bluetooth and Wi-Fi D2D communication variants. This paper discusses technologies such as firewalls and Intrusion Detection Systems (IDS) to solve these concerns and make D2D connections more secure and reliable for users.

In addition, [22] investigates a D2D network in which one device functions as an access point (D2D Relay) for the other devices connected to the network via WiFi Direct. In particular, the authors of [22] exploited a device linked to the D2D network to attack an Internet-hosted server. It has been observed that additional devices connected to the access point cannot connect to the Internet. In addition, a second assault has been introduced on a device hosting a server in the D2D network. It has been noticed that the server on the victim device is forced to restart; however, the other devices can still maintain Internet connectivity.

Through the exchange of resources in 5G D2D networks, the paper's authors [55] examine a potential DoS attack strategy for a double auction game. The auction game based on the DoS attack described in this work degrades service by preventing buyers and sellers from participating in the double auction. The report then describes a countermeasure for similar DoS attacks. The mathematical evidence of the proposed technique is presented alongside the corresponding emulation results.

In [56], authors use a variety of Machine Learning (ML) techniques, such as random forest, light GBM, XGBoost, and AdaBoost, to quantify detection accuracy in the context of DDoS and DoS attacks and examine their performance through extensive simulation. According to the gathered results, random forest improves the accuracy of both the Slowloris and CIC-DDoS2019 datasets. They also create a method for combining DDoS and DoS attack detection in binary classification Random Forests with a binary decision.

C. COMPARISON OF RELATED WORKS THAT UTILISED ML USED AT IDS

This section provides a table for comparing our approach with the other competitive techniques found in the literature. Thus, Table 1 shows the strengths or weaknesses of the examined approaches in the literature review compared to our approach. More specifically, the table provides an overview of different intrusion detection datasets, methods, and their respective accuracy rates. The datasets primarily pertain to Gateway-Sinkhole/Distributed/Cloud systems, and the network intrusions include Probe, DoS, Remote to Local (R2L), User to Root (U2R), Botnet, Infiltration, DDoS, etc. The table details whether the dataset and attacks are identified, the accuracy rates of various intrusion detection approaches, whether they are AI/ML-based, whether they can be used for D2D communication, and whether they can detect zero-day attacks.

Overall, the table compares and selects appropriate datasets and intrusion detection methods for various network system types. Also, most of the papers that are examined in the literature review section do not achieve the identification and classification of attacks that our paper achieves. Our paper achieves the identification of twelve known attacks and four unknown attacks. However, it can identify more unknown attacks where most of the papers identify one or at most three attacks. Additionally, the proposed approach is the only one that uses DHMLM. Only some utilise deep neural networks from the other approaches: the [20] and [50]. Also, regarding the dataset, this paper and [56] used a well-known dataset with an emulating dataset. Moreover, this paper and [56] support D2D communication. Furthermore, our approach is the only one supporting unknown/zero-day attacks. Finally, in terms of accuracy, the most accurate approaches with more than 99% in increasing order are i) this article with 99.07%; ii) the [56] with 99.4%; and iii) lastly the [48] with 99.79%. Even though our approach has less accuracy than the other approaches, it can identify far more known and unknown attacks with high and acceptable accuracy.

III. PROBLEM DESCRIPTION AND OBJECTIVE

The demand for faster and more efficient data transmission in network communications gave rise to 5G technology, which relies on D2D infrastructure. D2D communication shortens the link by eliminating the need for a Base Station (BS). This allows for increased independence from network infrastructure, paving the way for increased autonomy. D2D has been hailed as the solution for emergency response communications in a scenario with no network coverage. However, D2D communication exposes UEs to security threats such as DDoS attacks, SQL injection, and brute force attacks.

Even though this paper tackles various types of attacks, it focuses more on the DDoS type of attacks, which is considered one of the most common attacks in the commercial scenario [57]. DDoS attacks cause global outages and continue to threaten many forms of communication; they are not limited to D2D communications. Although some solutions have been proposed[4] and described in Section II-B, none have proven to be effective in D2D Communication Networks due to the hardware limitation of the mobile devices. Towards this end, the primary objective of this study is to detect common security threats in D2D networks, particularly DDoS attacks, by employing a DHMLM on a partially emulated data set.

A. ASSUMPTIONS

Our investigation is based on the following assumptions:

1) All emulation-related devices must be connected to the same wireless access point (WAP).
2) All devices used in the emulation setup are not susceptible to multiple attack types at any given time. Thus, only one type of attack occurs per time step on the devices.
3) CPU and RAM are sufficient for training and identifying the examined attacks.
4) Utilising sampling methods, the disparity in the generated data points for each attack signature is evened out before model training.
5) Low-to-moderate computational power simulates the real-world D2D infrastructure environment.
6) Due to the nature of the attack, no permanent damage is done to the affected devices. Battery depletion, device overheating, etc., are temporary symptoms that vanish quickly [47].

[4]Such as a shared secret key approach to prevent Man-in-the-Middle (MITM) attacks, which entails a commitment scheme to authenticate devices mutually. However, this is impractical due to the need for visual and verbal confirmation [58]. Compared to traditional WANs. The Wide Area Networks are the conventional network infrastructure primarily utilised in wireless networking. Such networks can span multiple Local Area Networks (LANs) and vast geographical distances. It heavily employs a base station or controller. WAN devices are predominantly heterogeneous and range from low to high computational power. The involved devices must therefore be connected to the same wireless access point, user privacy is not yet adequately protected on modern WANs.

TABLE 1. **Comparison of related works that utilised ML in the context of IDS in the open literature.**

Reference	Detection location Gateway-Sinkhole/ Distributed/ Cloud	Dataset The dataset is publicly available or NOT	Attacks Identified	Accuracy	Approach	AI/ML?	D2D?	Zero/Unknown Attack Identification?
[20]	Gateway	Knowledge Discovery and Data Mining (KDD) Cup 1999 Data(yes)	Probe, DoS, Remote to Local (R2L), User to Root (U2R)	Not mention	Intrusion detection using deep neural networks	yes	no	no
[46]	Gateway	KDDCup 1999 Data(yes)	Probe, DoS, Remote to Local (R2L), User to Root (U2R)	95.03%	Intrusion Detection using feature grouping based on linear correlation. coefficient with cuttlefish algorithm	yes	no	no
[48]	Gateway	NSL-KDD (yes)	Probe, DoS, Remote to Local (R2L), User to Root (U2R)	99.79%	Intrusion detection using feature selection techniques and J48 DT	yes	no	no
[49]	Gateway	Game Theory and Cyber Security Data(yes)	Botnet, Infiltration, Distributed denial of service (DDoS), Brute force	98.62%	Intrusion Detection using a novel data set for assembling	yes	no	no
[50]	Gateway	Modified Telecom Italia 2015 Data(no)	Silent Call, Signaling, SMS Flooding / SMS Spamming	greater than 91% for each cell	DDoS attack detection using Deep Rudimentary CNN for call detail record (CDR)	yes	no	no
[51]	Gateway	online monitoring and filtering of traffic Data(yes)	Scanning, Passive Scanning, Virus Remote to Local (R2L), Exploits, Denial of Service (DoS),	not mentioned	Intrusion Detection using levels of granularity using a fusion detection algorithm	yes	no	no
[52]	Gateway	CAIDA DDoS Attack2007 Data(on request)	attack flow only	not mentioned	DDoS attack detection using multiple kernel learning	yes	no	no
[56]	Gateway	Custom - CIC-DDoS2019 and Emulated Data (yes)	SYN, Slowloris	99.4%	Intrusion Detection using Random Forest, Extreme Gradient Boosting (XG Boost), Adaptive Boost (Ada Boost) and light gradient-boosting machine (LGBM)	yes	yes	no
This Paper	Can be used at Gateway-Sinkhole /Distributed/Cloud	Custom/Mixed - CIC-DDoS2019 and Emulated Data (yes)	Trivial File Transfer Protocol (TFTP), User Datagram Transfer Protocol (UDP) , Lightweight Directory Access Protocol (LDAP) , Simple Service Discovery Protocol (UDP), Man in the Middle (MITM) Network Basic Input/Output System (NetBIOS), Portmap, SYN Microsoft SQL Server (MSSQL), Dictionary, Are you dead yet ?, (RUDY) SQL Injection SQLi), GoldenEye, HTTP Unbearable Load King (HTTP)	99.07% with Deep Hierarchical Machine Learning Model	Intrusion Detection investigation using LSTM, DNN, Simple RNN and DHMLM	yes	yes	yes

7) No flows are lost or altered during the transmission of a Packet Capture (PCAP) file to a device with the CicFlowMeter application, which uses the secure Simple Mail Transfer Protocol (SMTP) protocol.
8) The unknown Zero-day attack is drawn from the same distribution as the generated attacks, **but it was not included in the training data set provided to the DHMLM**.

IV. SYSTEM DESCRIPTION

This section includes a detailed description of our system: it clarifies our system's entire end-to-end workflow setup and details about the investigated model's architecture (i.e., hierarchical, deep learning, and recurrent).

System Workflow:

The entire procedure, from feature selection to testing, is depicted in Fig. 1. The data from Packet Capture (PCAP) files converted to Comma-Separated Values (CSV) format are combined with those from the CIC-DDoS2019 data set. Additionally, the data are preprocessed by removing the records that have values of Not Any ("NAN") (called by Python as Not Any or NULL in other languages and they are fields of any type that have no value as a result) of any data type (i.e., int, float, complex). The reduced feature set is then sent as train data to the RNN, DNN, LSTM, and DHMLM for training purposes after being normalised with a mean of 0 and a standard deviation of 1. Next, parameter hyper-tuning to determine the optimal bias-variance trade-off testing is performed. The final step is detecting unknown attacks, which investigates and evaluates the neural network's ability to detect zero-day attacks.

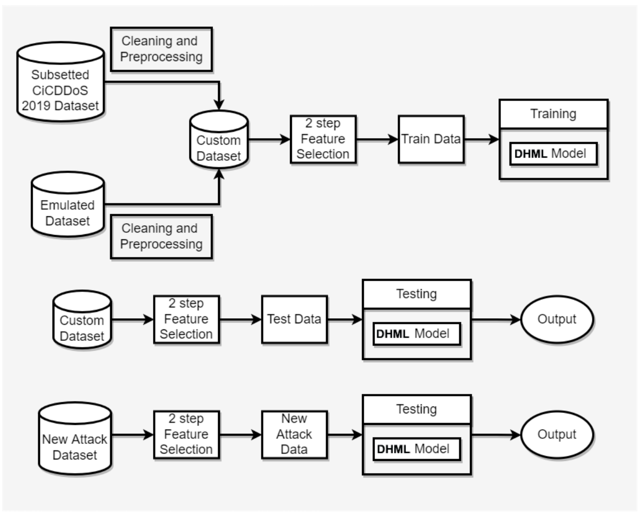


FIGURE 1. **Workflow of system.**

V. METHODOLOGY

This section describes the methodology employed in the proposed approach. First, hierarchical architecture of the model is shown, and then a detailed description of the emulation environment for attack generation is provided. Next, the data set generation process, along with the manipulation methodology, is described to demonstrate how: i) sampled and emulated attacks are combined; ii) the data set is improved

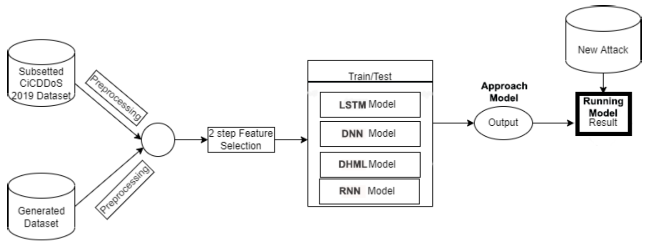


FIGURE 2. **Used comparative models.**

by cleansing; iii) the best feature set that will be used to train our model and identify an attack, is selected. The two-step feature selection method and experimental results for determining the optimal feature subset are described. Finally, the proposed Hierarchical Machine Learning Model's structure is presented.

Proposed Deep Hierarchical Machine Learning Model Architecture: In our approach, we consider a DHMLM/DHNN, a Recurrent Neural Network (RNN), a Long short-term memory (LSTM), and a Traditional Deep Neural Network Model (DNN) to scale our model (as shown in Fig. 2). Starting with the RNN, the RNN classifies an attack signature using a many-to-many architecture with a 13-way soft-max classification output layer. Continuing with the LSTM, the LSTM classifies an attack signature with a 13-way soft-max classification output layer. Finally, the traditional DNN classifies an attack signature using a 13-way soft-max classification output layer. Overall, the results of the aforementioned approaches, however, result in less precision than the DHMLM. The Hierarchical Machine Learning model attempts to discover and observe hierarchical patterns within the data set for decision-making purposes. Using three labels, the data set establishes the specified hierarchy.

As shown in Fig. 3, the DHMLM comprises three network levels. Each label is used to classify attacks at the corresponding neural network level. The initial distinction is between hostile and benign values. Therefore, this level requires the highest degree of precision to prevent the accumulation of errors at deeper levels and to mitigate the problem of false positive alarms that plague the majority of IDSs. Only if the first level classifies the data point as an attack signature is the second level activated. The model distinguishes between a DDoS signature and other common attack signatures at this level. The third and final level is activated if the second level assigns the data to the DDoS classification. This level aims to distinguish between various DDoS attack signatures. Regarding computational complexity, the worst-case scenario involves the network packet being a DDoS attack, while the best-case scenario involves a benign packet.

A. EMULATION ENVIRONMENT

In this section, we describe the emulation environment setup with the network design and the hardware characteristics of each device. Thus, in Figure 4, the creation and formation of the D2D communication network are implemented when the D2D device selects to share its bandwidth and select the D2D-Relay[6] transmission mode (as shown in [59]) and when the rest of the devices select to connect to this device as D2D-Clients.[7] The setup is instantaneously ready using the code implemented for DAI Framework (as shown in [59]).

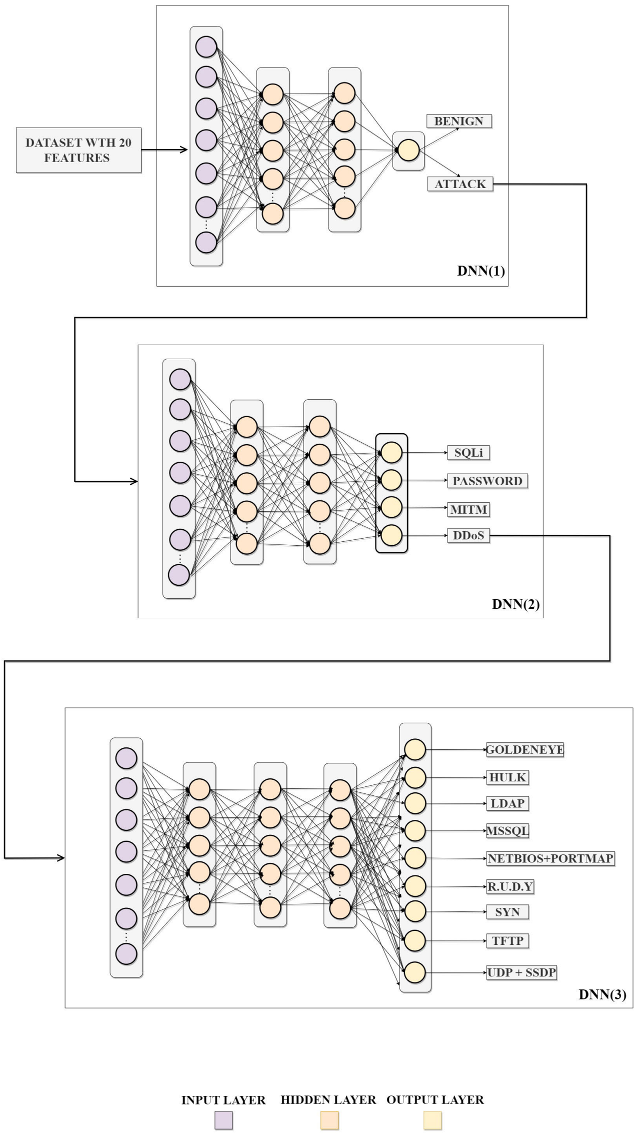


FIGURE 3. **Hierarchical machine learning model.**

In our emulation configuration, as shown in Figure 4, we have the following participation devices: i) three mobile devices (Samsung Galaxy S23s Ultra) running Android 13.0, one of which is acting as a D2D-Relay and the other as a D2D-Client. Note that both of them are victims of the attackers; ii) three Android 12.0 tablet devices (Samsung Galaxy Tab S8+) are all victims of the attackers; iii) two personal computers,

[6]One of the D2D Devices is connected to a BS or Access Point and provides access, by sharing its bandwidth, to another D2D Device/Devices.

[7]There are the devices that connect to the device that is sharing its bandwidth.

one of them installed (named PC1) at a home network with Windows 11 (with an Intel Core i7-8550U processor, 16GB of RAM and Intel UHD 620 graphics card) and Virtual Box hypervisor having as a virtual machine the Kali Linux 2020 (with an Intel Core i7-8550U processor, 8GB of RAM and Intel UHD 620 graphics card) [60] and within the same mobile network of the same operator to the mobile devices and tablets and acts as an attacker (called PC1/VM1). The other is in the same D2D communication network created by the D2D relay. At the second personal computer (named PC2) under the D2D communication network, we installed Oracle virtual box hypervisor on a Windows 11 machine (with an Intel Core i7-8550U processor, 16GB of RAM, and Intel UHD 620 graphics card). The reason is to create two virtual machines. The first virtual machine has Kali Linux 2022 (with an Intel Core i7-8550U processor, 8GB of RAM, and Intel UHD 620 graphics card) in order to attack the D2D communication network from the inside (called PC2/VM1). The other has Metasploitable 2 (with an Intel Core i7-8550U processor, 8GB of RAM, and Intel UHD 620 graphics card) [61] to be the vulnerable victim within the network (called VM2) using a NodeJS server [62] as HTTP server; and lastly, iv) A Windows 11 (with 16GB RAM, AMD Ryzen 9 5900HS, and NVIDIA GeForce RTX 3060 GPU) personal computer (called PC3) with Jupyter Notebooks and a cloud subscription to on Google Colab Pro with 25GB RAM and 16GB GPU RAM, the DNN, and DHMLM were trained and tested on this cloud subscription. Moreover, we have also installed Oracle Virtual Box and augmented it with a virtual machine with Ubuntu 22.04 and Node JS server (with 8GB RAM, AMD Ryzen 9 5900HS, and NVIDIA GeForce RTX 3060 GPU) on this personal computer (PC3/VM1). The attacks were emulated on four distinguished virtual devices, three mobile devices, and three tablets that are connected to a D2D communication network. MITM and SQLi attacks were launched from a PC2/VM1 Kali Linux 2020 Virtual Machine against a PC2/VM2 Metasploitable 2 Virtual Machine that hosted a web server and towards the D2D-Client mobile and tablet devices. Continuing with the rest of the attacks, a PC1/VM1 Kali Linux 2020 Virtual Machine generated the HULK, HTTP Flood, GoldenEye, and RUDY attacks towards the internal PC2/VM1 Kali Linux 2020 Virtual Machine and the D2D-Relay mobile device. The MITM and SQLi attacks were launched from a PC2/VM1 Kali Linux 2020 Virtual Machine against a PC2/VM2 Metasploitable 2 Virtual Machine that hosted a web server. Finally, a PC2/VM1 Kali Linux 2020 Virtual Machine attacked a Node JS server on the PC3/VM1 Ubuntu 22.04-powered victim virtual machine. Continuing on the packet sniffing and model building, the Wireshark, a tool for packet sniffing, was used to capture network packets during the attacks. The ".pcap" extension files were then converted to comma-separated values (CSV) files with the ".csv" extension using CICFlowMeter 3.0, a generator and analyser of network traffic flow on the Windows 11 where the DNN and DHMLM were trained and tested.

TABLE 2. Emulated attacks distribution for custom dataset creation.

Class	Attack	Number of Data points
Dictionary	Dictionary	4,638
GoldenEye	DDoS	47,995
H.U.L.K	DoS	285,762
MITM	MITM	8,618
R.U.D.Y	DoS	60,877
SQLi	SQLi	46,412
Total		454,302

B. UNIFIED DATASET CREATION FOR MODEL TRAINING AND COMPARISON

This section provides the process of generating the resulting joined Data set among a custom-created Dataset by us with the CIC-DDoS2019 Dataset. More specifically, the dataset creation process is executed using our custom dataset joined with the CIC-DDoS2019 dataset, targeting the creation of a unified dataset for model training and comparison among our investigated approaches. Furthermore, the current and existing dataset of CIC-DDoS2019 selection is performed based on the criteria of being widely recognised, extensively utilised, and highly adaptable for D2D communication. Finally, the reasons for creating a merged unified dataset are stated in this section.

Thus, to start with, this subsection provides the size proportion of sampled data points used and an in-depth explanation of how the data points are gathered from the attacks that were emulated under the D2D communication network, along with their proportions and graphical representations. More specifically, Table 2 shows the emulated attacks.

1) DATASETS CREATION PROCESS

In this section, we provide the process of creating the custom dataset and joining with the CIC-DDoS2019 subset, which is created by selecting a large set of elements from the CIC-DDoS2019 Evaluation Dataset (95% from the CIC-DDoS2019 dataset related to the attacks we target to identify). The emulated data set was generated using the following process: Firstly, the Attack Device is prepared by installing Virtual Box, a Kali Linux image to run on top of it, and the Kali Tools required to emulate the attacks, e.g. GoldenEye, HULK, R.U.D.Y, xHydra, #RefRef, and Ettercap are some tools that we use in the investigation. Next, the Victim Device is prepared by installing Metasploitable 2 [63] or Ubuntu 22.04 with a NodeJS server. Note that Metasploitable 2 is a Virtual Machine that hosts a vulnerable web server in order to allow penetration testing, and the NodeJS server hosts a simple web page on an arbitrary port. Then Wireshark is installed and used on another device that is in monitor mode to sniff all the traffic of the D2D Relay device. Wireshark is software that monitors network packets and captures them as a ".pcap" extension file. Continuing, the ".pcap" extension file is sent to a device containing the CIC-Flowmeter (CicFlowMeter) [64] application that transforms ".pcap" extension files

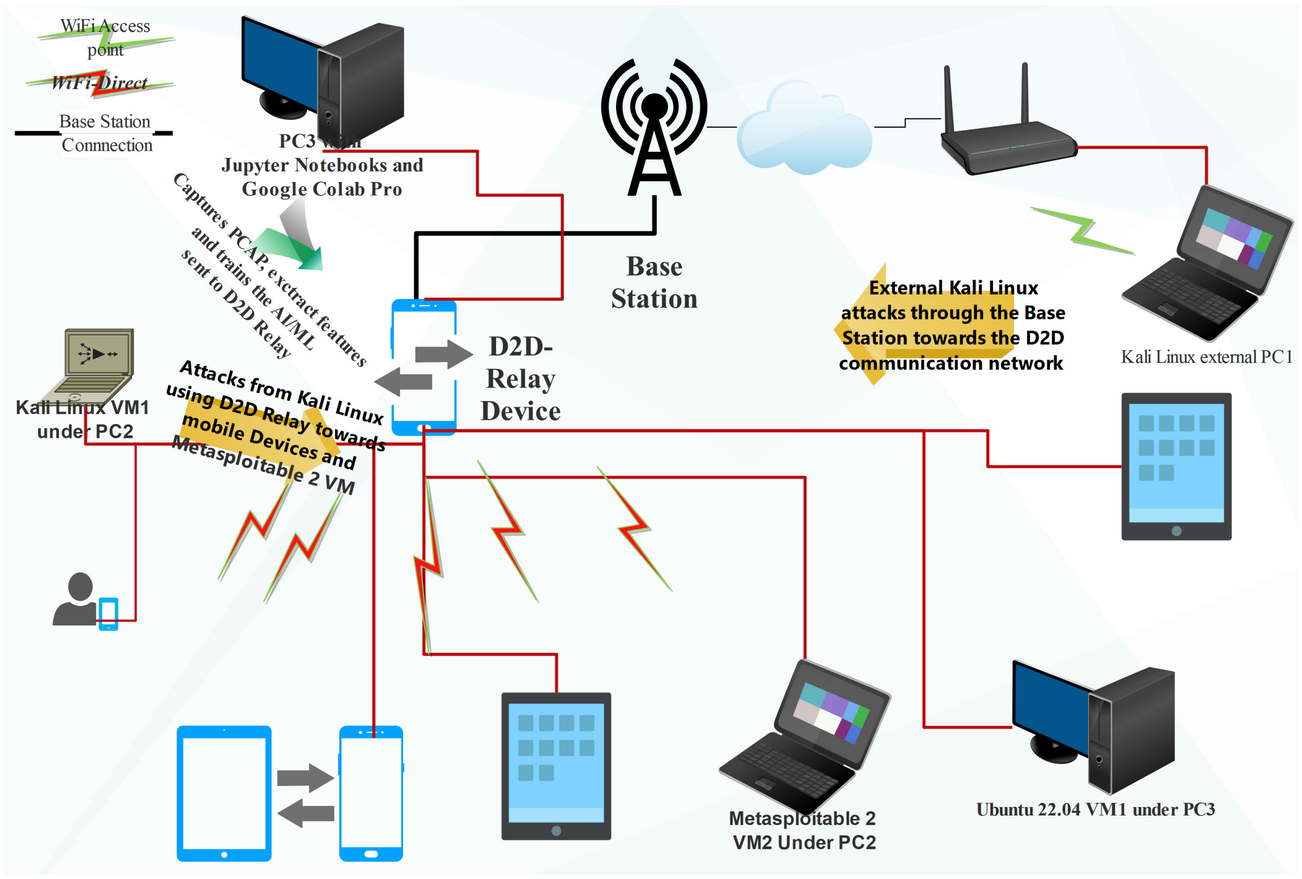

WiFi Access point
WiFi-Direct
Base Station Connnection
PC3 with Jupyter Notebooks and Google Colab Pro
Captures PCAP, extract features and trains the AI/ML sent to D2D Relay
Base Station
D2D-Relay Device
External Kali Linux attacks through the Base Station towards the D2D communication network
Kali Linux external PC1
Kali Linux VM1 under PC2
Attacks from Kali Linux using D2D Relay towards mobile Devices and Metasploitable 2 VM
Metasploitable 2 VM2 Under PC2
Ubuntu 22.04 VM1 under PC3

FIGURE 4. D2D communication network where emulation run.

into ".csv" extension files to mimic the attacks' format in the CicDos2019 data set with 87 features.

Note that the tools mentioned above are used to launch an attack on the web server hosted on the victim's device for a given period. This eventually leads to the web server's collapse due to the DoS/DDoS and other attacks crippling the web server and exhausting the resources. The following paragraphs show the tools used for specific attacks and how they are utilised in the emulation set-up in order to attack the victim device (which is the Metasploitable 2 or the Ubuntu NodeJS). Moreover, this paragraph provides a table with the attack records used from the CIC-DDoS2019 dataset. A more in-depth description of the tools can be found in Section II-A4. Thus, the attacks realised at our emulation D2D communication network are the following:

- The Man in the Middle attack (MITM), depicted in Fig. 5, was emulated after executing an open-source network security tool called Ettercap and by establishing a new connection targeting the MITM in Local Area Networks and resulting in password capturing and eavesdropping between the attacker virtual machine towards the Metasploitable 2 web server.
- The HTTP DoS attack was emulated using the R.U.D.Y tool to auto-detect web forms on the victims' website to allow the attacking user to decide which fields to exploit, utilise proxies and execute cookie-based session persistence, targeting the disability of the victim device's hosted web site via HTTP server to serve clients. The emulation setup is shown in Fig. 6.
- DDoS attack using open-source GoldenEye tool, which is a tool that allows users to create legitimate HTTP requests using a botnet. In our case, the Kali Linux 2020 tool attacked the NodeJs Kali Linux 2020 victim

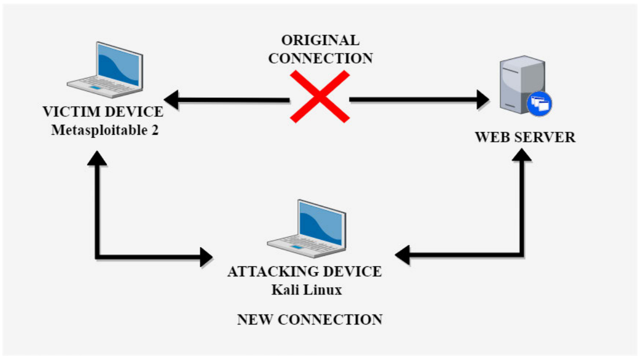


FIGURE 5. Man in the middle attack emulation setup.

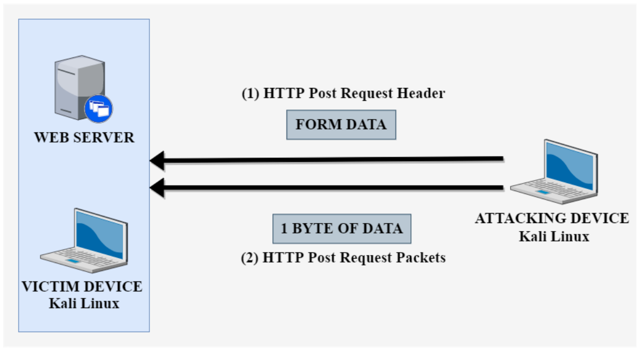


FIGURE 6. Are You dead yet (RUDY) attack tool emulation setup.

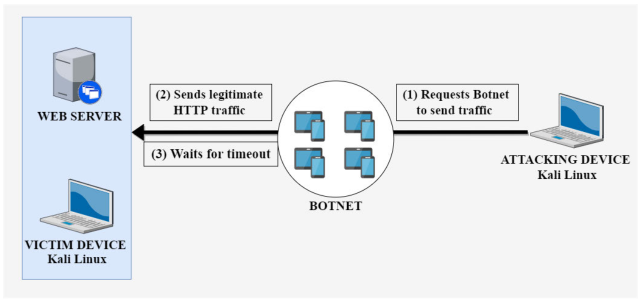


FIGURE 7. GoldenEye attack tool emulation setup.

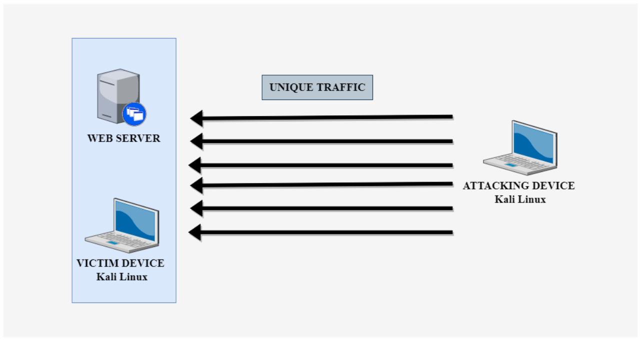


FIGURE 8. HULK tool emulation setup.

server and brought it down. The emulation setup is shown in Fig. 7.

- The DoS HTTP attack is implemented using HULK tool through an obfuscation technique that it masks the Hulk server generating the attack requests. It modifies the URL in the request to look legitimate, targeting to overwhelm the web server hosted on the victim device with many requests, as depicted in Fig. 8.
- For password attack, the xHydra tool is used to make a dictionary attack of usernames and passwords in order to attempt to crack the password on the victim's device, as shown in Fig. 9. For the configuration, a cooldown of 30 seconds after every 16 attempts was used to avoid detection by the target authentication system.
- The SQL Injection (SQLi) Attack is realised with the use of #Refref tool. As shown in Fig. 10, it leverages the

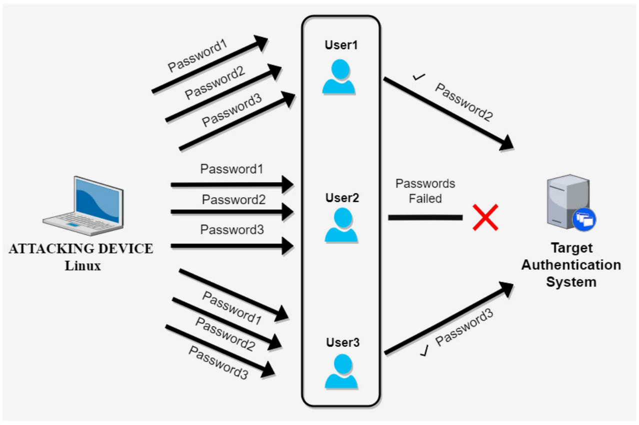


FIGURE 9. xHydra tool emulation setup.

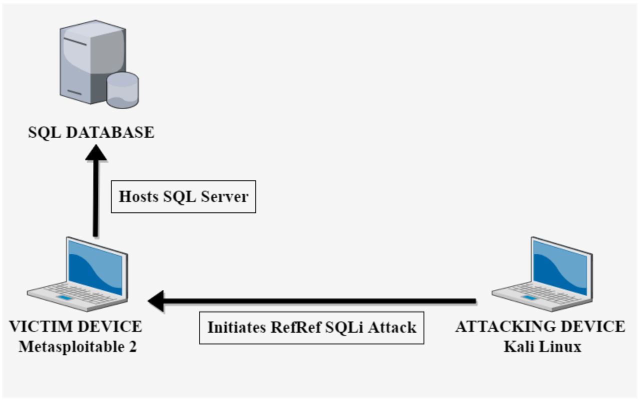


FIGURE 10. #Refref tool emulation setup.

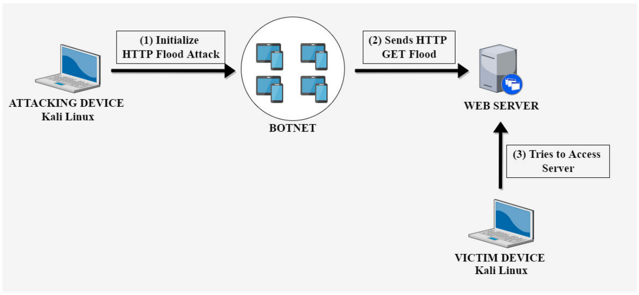


FIGURE 11. HTTP flood attack tool emulation setup.

My-SQL (MySQL) vulnerability in a dummy website on Metasploitable 2 to perform SQL injection involving the Benchmark() function, allowing for repeated execution of an expression to cripple the website.

- For the DDoS attack at HTTP protocol on GET and POST requests, we used the FLOOD tool that internally, the botnet floods overwhelm the server with GET (in our case, but it can send POST also) requests so that the web page cannot be served to the client. Note that this attack is not included in the training set and is used as the Zero-day attack to test our model. The emulation setup is shown in Fig. 11.

TABLE 3. Sampled attacks distribution with the use of CIC-DDoS.

Protocol/Class	Attack	Number of Data points
TFTP	DDoS	199,764
UDP+SSDP	Dos, DDoS	398,538
DDoS	LDAP	180,578
NetBIOS+Portmap	DDoS, DDoS	405,787
SYN	DDoS	198,373
MSSQL	DDoS	1,242
Benign	N.A.	7,412
Total		1,391,694

Table 3 shows data points sampled from the CIC-DDoS2019 data set.

2) JOINED DATASET CREATION

This section shows how the joining of both datasets is achieved, resulting to the unified dataset to be used for training the IDS.

To build an IDS that can detect many attacks, some attacks were sampled from the CIC-DDoS2019 data set [15] and were merged with the attacks we emulated. The attacks from the CIC-DDoS2019 data set contained 86 features, and those that were sampled are based on the following protocols/classes TFTP, UDP, SSDP, LDAP, NetBIOS, Portmap, SYN, and MSSQL (using more than 95% of the dataset). The benign values, which are the records in the dataset that show that no attack is executed at the current time, were also sampled from this data set. From [65], it can be seen that for this set of features, the attacks Portmap and NetBIOS are highly correlated with a correlation of 0.46 value. The SSDP and UDP attacks correlate by 0.68 value. The accuracy of the Neural Network drops as it misclassifies these attacks to 30%. It is worth mentioning here that the attacks from the CIC-DDoS2019 data set were undersampled due to computational constraints.

The solution followed in the attack emulation process is similar but not identical to the one proposed in [65]; i.e., similarly to Portmap and NetBIOS, the SSDP and UDP were merged. The emulated attacks were executed on the following protocols: HULK, R.U.D.Y, MITM, SQL injection, GoldenEye, and Password. These attacks contained 83 features, which is three features less than those contained in the CIC-DDoS2019 data set. The reason is that the CiCFlowMeter did not generate three features in the sampled attacks: the 'SimilarHttp', 'Inbound', and 'Fwd Header Length.1'.

The emulated and sampled attacks were then merged. The resulting data set included 1,845,996 rows. Before training, the data points in the various classes were balanced to prevent skewed results. The synthetic Minority Oversampling Technique (SMOTE) was used to oversample the classes with fewer data points. SMOTE uses the k-nearest neighbour algorithm to create synthetic data points until the minority classes are in the same proportion as the majority class [66]. Overall, the joined data set was prepared by sampling attacks from CIC-DDoS2019 data set [15], [67] and by generating attack signatures with the help of common attacking tools on the Kali Linux Operating System (such as DDoS GoldenEye tool). Thus, finally, the resulting data set had a total of 1,845,996 data points.

3) THE REASONS FOR CREATING A JOINED DATASET WITH A PRESTIGIOUS WELL-USED DATASET OF "CIC-DDOS2019"

In this subsection, the reasons for creating a joined dataset with a prestigious well-used dataset of "CIC-DDoS2019" are shown and they are the following:

- **Diversified data source**: The incorporation of a bespoke dataset has the potential to augment the breadth of data sources, encompassing a broader spectrum of network configurations, traffic behaviours, and instances of malicious attacks that may not be comprehensively represented by the CIC-DDoS2019 dataset alone including the investigation of a D2D communication network. The integration of distinct datasets can augment the comprehensiveness and scope of understanding regarding Distributed Denial of Service (DDoS) attacks and their various manifestations, along with other investigated security attacks.
- **Real-world relevance**: A tailored dataset can be acquired from a specific setting or context relevant to the research being undertaken, such as a particular sector or organisation. The utilisation of this methodology has the potential to augment the authenticity of scenarios employed for the purposes of testing and analysis, hence amplifying the relevance of the findings to real-world situations. The dataset we have generated and it is merged with the well-used dataset is based on the D2D communication network.
- **Data imbalance mitigation**: The matter holds significant importance within the framework of the CIC-DDoS2019 dataset and other readily available datasets. There is an observable discrepancy within the dataset, namely in representing some classes, such as assault kinds, which may require more attention to ensure a more equitable distribution. One possible strategy for addressing this problem entails constructing a customised dataset focusing on the underrepresented categories. The performance and generalizability of the model can be improved by increasing the number of samples associated with these classes.
- **Anomaly detection**: Integrating a custom dataset with CIC-DDoS2019 can be advantageous for training models in anomaly detection, a critical task in identifying emerging attack trends. The inclusion of our manual custom data has captured anomalies that may need to be better represented in the original dataset (i.e., SSDP attacks, LDAP attacks). Moreover, our custom data set can identify attacks not included in the original dataset (i.e., MITM attacks, Password attacks).
- **Enhancing Model Generalisation**: The inclusion of diverse network traffic patterns and attack scenarios

derived from our customised dataset significantly enhanced the overall ability of machine learning models to generalise. This has the ability to improve the effectiveness of the resulting models in mitigating new attacks and adapting to various network conditions.

- **Feature Engineering and Selection**: In the realm of D2D attack detection, it may be relevant to include supplementary features or attributes in a customised dataset for the purpose of feature engineering and selection. The incorporation of datasets enables the utilisation of a broader range of attributes, hence enabling improved techniques for engineering and selecting features.
- **Benchmarking and Evaluation**: The creation of a merged dataset allows the benchmarking and comparative evaluation of many models utilising a larger and more comprehensive dataset. This methodology enables the identification of models that demonstrate superior performance across a wide range of data sources and scenarios.
- **Customised Research Objectives**: The CIC-DDoS2019 dataset does not sufficiently address an individual's research objectives, such as studying different D2D communication attack behaviours in specific scenarios or evaluating the effectiveness of particular countermeasures, so a customised dataset is created to merge with the CIC-DDoS2019 and meet those additional objectives.
- **Privacy and security**: The challenges pertaining to privacy and security that arise from the utilisation of real-world data can be addressed by gathering a customised dataset within a regulated setting. The dissemination or publication of research findings carries significant importance within academic settings.
- **Advancing the Field**: Individuals have the opportunity to contribute to the advancement of D2D attack detection and mitigation techniques, as well as the present knowledge and practices in this sector; this research offers an improved dataset to the academic community. The custom dataset developed exhibits considerable potential as a valuable resource for future scholars to employ and extend in their own inquiries.

C. FEATURE SELECTION

This section provides information on selecting the correct features to train the ML model. Statistically-based feature selection procedures involve evaluating the association between each input variable and the target variable using statistics and selecting the input variables with the strongest association with the target variable. Even though the choice of statistical measures is dependent on the data types of both the input and output variables, these methods can be swift and effective. Feature selection in predictive model construction is the process of minimising the number of input variables. Reducing the number of input variables can reduce the computational cost of modelling and, in some cases, improve the model's performance [68].

TABLE 4. HML accuracies on feature subsets.

Number of Features	Level 1	Level 2	Level 3	Overall Accuracy
5	84.80%	89.73%	94.37%	89.63%
9	96.06%	91.50%	95.82%	94.46%
10	90.08%	87.86%	96.32%	91.42%
11	96.34%	94.06%	97.63%	96.01%
15	98.34%	98.96%	97.30%	98.20%
20	99.36%	99.23%	98.58%	99.07%
25	98.89%	99.06%	98.00%	98.65%
30	98.43%	99.42%	98.16%	98.67%

A two-step feature selection method is proposed to select the most suitable feature set to train our model and identify an attack. This method consists of a correlation-based feature selection followed by the Mutual Information-Based Feature Selection (MIFS) Algorithm [69]. More specifically, the correlation matrix is a feature selection algorithm that employs correlation to obtain features with high correlation to one another and eliminates one of them because they have the same effect on the dependent variable. Also, Mutual Information-based (MI) assesses any arbitrary dependence between two variables. This feature selection algorithm uses MI to rank the best features within a given set of features.

After removing highly correlated features, the correlated feature method reduces the merged data set from 83 features to 48 features. Mutual information gain is used to rank and select feature subsets from this feature set. The DHML was trained on multiple subsets of the rated features to determine the subset that produced the highest accuracy for the model. Using a subset of the top 20 features indexed by the MIFS algorithm yielded the highest accuracy, and this subset was utilised for our final data set. Table 4 shows the experiments on different feature subsets, while Table 5 shows the optimal feature subset.

Note that our feature selection techniques are selected to achieve optimised attack identification of highly similar application attacks (i.e., NetBIOS and Portmap attacks), as well as protocol attacks (i.e., UDP and SSDP) [70].

D. WORK FLOW/ARCHITECTURAL DESCRIPTION OF THE INVESTIGATED APPROACHES

This section provides an architectural description of the investigated models (i.e., DHMLM, DNN, RNN, LSTM).

1) THE DESCRIPTION OF THE ARCHITECTURE AND OF THE HIERARCHICAL LEVELS OF DHMLM

As shown in Fig. 3, the architecture of the proposed hierarchical machine learning model (DHMLM) is described in this section. The implementation aims to localise the learning of various aspects of the data to specific network attacks.

Thus, in our architecture, the DHMLM consists of three layers; each layer comprises a DNN that learns one distinct level of the hierarchical data. The reason for selecting a hierarchical structure model is that it provides greater

TABLE 5. Optimal feature subset.

Feature Name	Description
Source Port	Port number used by source program
Destination Port	Port number used by destination program
Protocol	Protocol used by the program
Flow Duration	Duration of the flow in Microsecond
Total Fwd Packets	Total packets in the forward direction
Total Backward Packets	Total packets in the backward direction
Total Length of Fwd Packets	Total size of the packet in forward direction
Fwd Packet Length Min	Minimum size of the packet in forward direction
Bwd Packet Length Max	Maximum size of the packet in backward direction
Flow IAT Mean	Mean time between two packets sent in the flow
Flow IAT Std	Standard deviation time between two packets sent in the flow
Flow IAT Min	Minimum time between two packets sent in the flow
Fwd IAT Total	Total time between two packets sent in the forward direction
Fwd IAT Mean	Mean time between two packets sent in the forward direction
Fwd IAT Min	Minimum time between two packets sent in the forward direction
Fwd Header Length	Total bytes used for headers in the forward direction
Bwd Header Length	Total bytes used for headers in the backward direction
Bwd Packets/s	Number of backward packets per second
Init Win bytes backward	The total number of bytes sent in the initial window in the backward direction
NFlow Packets/s	Number of flow packets per second

flexibility and customisation than a DNN with a shallow structure. Because when only a shallow DNN can identify benign traffic, it constitutes most of any real-time traffic flow. More specifically, by restricting the number of nodes and layers, these shallow DNNs are created to be less complex than conventional DNNs to reduce computational power consumption. This is especially useful when the UEs involved in D2D communication have limited computational capability. The following description applies to the DHMLM's layers:

- The first layer (DNN(1)) was trained to classify a packet as an attack or benign. DNN(1) has twenty input nodes and two hidden layers with seven and three nodes and reLU as the activation function. The output layer consists of one node with a sigmoid activation function.
- The second layer (DNN(2)) classifies the packet as a DDoS attack, Dictionary attack, MITM attack, or SQL injection. It has twenty input nodes and two hidden layers consisting of six nodes and four nodes, respectively. The activation function used for the hidden layer was Rectified Linear Unit (ReLU) [71]. The output layer contains four nodes with a softmax activation function.
- The third layer (DNN(3)) classifies the packet as TFTP, UDP/SSDP, LDAP, NetBIOS/Portmap, SYN, MSSQL, R.U.D.Y, H.U.L.K, and Goldeneye. DNN(3) has an input layer of twenty nodes. There are three hidden layers of nine nodes each. The reLU activation function is used for both the hidden layers. The output layer consists of nine nodes with a softmax activation function.

The data's train-test split is of the seven to three ratio (7:3 or 70% training vs 30% validation). All three DNN networks are trained using the Adam optimiser for 20 epochs in batches of 10000. The loss function used to train the first network was binary cross-entropy, and categorical cross-entropy was used for the other two networks. The following formulas as used for the calculation of the Binary Cross Entropy (BCE) and Categorical Cross Entropy (CCE) [72]:

$$BCE = -\frac{1}{n}\sum((y_a \times log(p_a) + (1 - y_a) \times log(1 - p_a))) \tag{1}$$

y_a represents the actual class label, p_a denotes the predicted label and n is the number of instances. p_a equals the probability of class 1, and 1 - p_a will be the probability of class 0.

$$CCE = -\sum_{i=1}^{n}(y_i \times logp_i) \tag{2}$$

y_i represents the actual class label, p_i denotes the predicted label, and n is the number of classes.

2) THE DESCRIPTION OF THE ALTERNATIVE APPROACHES ARCHITECTURES THAT ARE COMPARED TO THE DHMLM

In this paragraph, we show the workflow description of the comparative approaches to the DHMLM approach. The architectures of the comparative approaches are the following:

- Deep Neural Network (DNN): The DNN consists with five hidden layers and twelve nodes per layer. The twenty selected features (see section V-C) were provided as input. All hidden layers were activated with the reLU function. The output layer consisted of thirteen nodes, with twelve nodes representing the twelve attack classes (as shown in 2 and 3 and analysed in Section II-A4) and one node representing the benign packet class.
- A Recurrent Neural Network (RNN)[8]: The RNN consists of twenty inputs and a simple RNN layer. The twenty selected features (see section V-C) were provided as input. The hidden output dense layer is activated with the reLU function. The output layer was comprised of thirteen nodes, with twelve nodes representing the twelve attack classes (as shown in Tables 2 and 3 and analysed in Section II-A4) and one node representing the benign packet class.
- A Long Short-Term Memory (LSTM): The LSTM consists of twenty inputs and a simple LSTM layer. The twenty selected features (see section V-C) were provided as input. The hidden output dense layer is activated with the reLU function. The output layer consisted of thirteen nodes, with twelve nodes representing the twelve

[8]RNNs are called recurrent because they perform the same task for every element of a sequence, with the output being depended on the previous computations by adjusting its weights and biases through a process called backpropagation through time (BPTT).

attack classes (as shown in 2 and 3 and analysed in Section II-A4) and one node representing the benign packet class.

For all the models, the hyper-parameters were chosen to target the maximum performance improvement of the comparable approaches in the paper (e.g., DNNs) in order to be fair with all the models. Thus, these selections will make the models compatible with our proposed model in terms of accuracy.

Overall, these architectural implementations of our models aligned them to achieve the required results and enabled us to compare the effectiveness of our models in terms of accuracy, training time, power consumption, number of connections, and retraining time. In addition, the performance of the proposed model's first layer on a zero-day attack data set is observed.

VI. PERFORMANCE EVALUATION

This section presents an overview of the metrics used to validate the models and evaluate the results extracted from the approaches proposed in this paper to select the best approach. Continuing the investigated approaches, results are then presented and the best-selected approach of the investigated approaches is then compared with other recent related works like Neural Networks Multi-Classifier [65], Optimum K-Nearest Neighbour (OKNN) [73], and EagerNet [74]. For the evaluation process, all models are trained using a joined dataset among the CIC-DDoS2019 and a custom-generated dataset (as shown in Section V-B). This joined dataset acts as a unified dataset for comparison. The reason is to be sure that all models are equally handled and compared. Also, we have compared them (as shown in Section VI-B2) in terms of specific metrics as shown in Section VI-A2 (i.e., accuracy).

A. OVERVIEW OF THE METRICS USED FOR THE VALIDATION AND THE EVALUATION OF THE INVESTIGATED APPROACHES

In this section, we show the metrics we used in order to validate the investigated models and evaluate them. To validate and check the investigated models in terms of over-fitting and under-fitting but also the metrics we used to evaluate them and make them comparable, targeting the identification of the best approach among the investigated. Also, we compare our models' performance to those of other recent multi-class classifiers found in open literature with the use of the following metrics.

1) K-FOLD CROSS-VALIDATION

The K-fold cross-validation is used to validate and check the investigated models. The K-fold cross-validation with $k = 5$ was used to ensure the model was generalised well and could obtain the same metrics with different subsets of data points. K-fold cross-validation splits the data set into k-folds of uniform size. Following this, the first $(k-1)$ folds are utilised in training the model, and the average accuracy is obtained. The k-th fold is used to test the model obtained.

2) METRICS USED

To evaluate our models, we must compare their performance with the other models. Thus, this section provides the metrics used to measure the models' performance. In addition to the following metrics, the efficacy of the first layer of the proposed model on a data set of zero-day attacks is observed. Furthermore, to the metrics covered in this section, we employ training time, power consumption, connections, and retraining time.

a: CONFUSION MATRIX

The metrics used for evaluation were Accuracy, Precision, Recall, and F1 score [75], which are defined as follows:

$$Accuracy = \frac{t^p + t^n}{t^p + t^n + f^p + f^n} \tag{3}$$

$$Precision(P_r) = \frac{t^p}{t^p + f^p} \tag{4}$$

$$Recall(R_c) = \frac{t^p}{t^p + f^n} \tag{5}$$

$$f1score = 2 \times \frac{P_r \times R_c}{P_r + R_C} \tag{6}$$

The True Positive Rate (TPR) and False Positive Rate (FPR): The true-positive rate, also known as sensitivity, recall, or probability of detection, reflects the likelihood of an attack resulting in a real alert. The false-positive rate, also known as the likelihood of a false alarm, is equal to (1 specificity).

$$TPR = \frac{t^p}{p} = \frac{t^p}{t^p + f^n} = 1 - FPR \tag{7}$$

$$FPR = \frac{f^p}{p} = \frac{f^p}{f^p + t^n} = 1 - TPR \tag{8}$$

where t^p = True Positives (a result of an experiment that accurately identifies the presence of an assault), f^p = False Positives (a result of an investigation that accurately demonstrates the lack of an attack), t^n = True Negatives (an experimental result which wrongly indicates an attack), f^n = False Negatives (an experimental result which wrongly indicates a non-attack).

b: AREA UNDER CURVE (AUC) - RECEIVER OPERATOR CHARACTERISTIC (ROC) (CALLED AUC-ROC) CURVE/FIGURE METRIC

Another metric we utilise is the AUC-ROC Curve metric, the AUC-ROC is a graphical representation of the true positive rate (TPR) versus the false positive rate (FPR) at various classification thresholds. When the dataset is asymmetrical, i.e., one class is more prevalent than the other, the AUC-ROC metric is beneficial. In addition, it provides a single value that summarises the performance of a model across all possible classification thresholds, making it a popular metric across many fields. The AUC-ROC curve demonstrates a model's ability to differentiate between positive and negative classes by plotting the probability of a positive class as the threshold is varied between 0 and 1. A model with a higher AUC-ROC score has enhanced classification performance, indicating that it can better distinguish between positive and negative samples [76]. More specifically, the AUC quantifies the complete two-dimensional area beneath the ROC curve and represents the degree of differentiation between the two classes. The AUC of a perfect classifier is 1, while the AUC of a weak classifier is less than 0.50. The AUC-ROC curve is a valuable visualisation and comparison tool for classifier performance. A model with a greater AUC is more effective than one with a smaller AUC [76], [77].

B. RESULTS

In this section, we begin by analysing the performance results of each investigated method. Then, based on their precision and degree of difficulty, we select the approach that best fits our needs. Consequently, we compare the computational complexity and accuracy of various models, including DHMLM, DNN, RNN, and LSTM, with DHMLM being the most accurate model for solving the problem in this study. Next, using the unified dataset (shown in Section V-B), this section compares the competitive approaches found in the open literature with the selected investigated paper approach, which is DHMLM. So, we compare the performance of our model to other works and present the results of the experiments conducted. Continuing, we examine the identification of zero-day/unknown attacks employing our selected technique of DHMLM and demonstrate the capability of our model to identify unknown attacks. Finally, this section discusses the overall findings.

1) THE PERFORMANCE OF EACH APPROACH

This section provides our findings in terms of results for our investigation approaches.

a: RESULTS OF THE DHMLM

For DHMLM, the training accuracy and loss, along with validation accuracy and loss concerning the number of epochs for each layer (The first layer in our Deep Neural Network is represented as DDN(1), the second layer as DNN(2) and the third layer as DNN(3)), are given in Fig. 12, Fig. 13 and Fig. 14. Thus, Figures 12, 13 and 14 illustrate the training and validation performance of the DHMLM's three layers during their training and validation phase. The first subplot displays the training and validation accuracy of the model, while the second subplot displays the training and validation loss. Observing the first (Figure 12), second (Figures 13), and third figures (Figures 14) representing the layers in the Deep Hierarchical Machine Learning Model, along with their respective level representations. Notice that we observed a similar pattern in all figures at all levels to that of the first figure. More specifically, when the training accuracy steadily improves

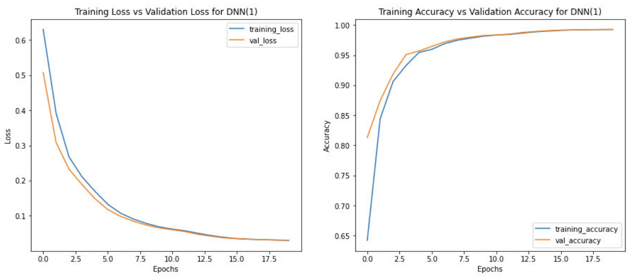


FIGURE 12. Training & validation performance DNN(1).

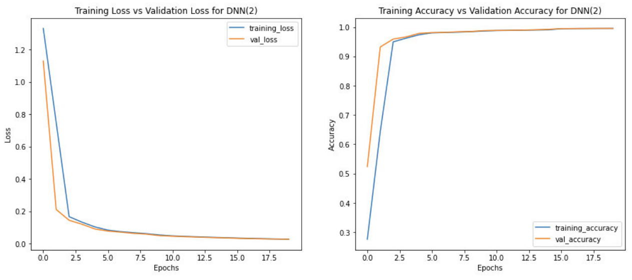


FIGURE 13. Training & validation performance DNN(2).

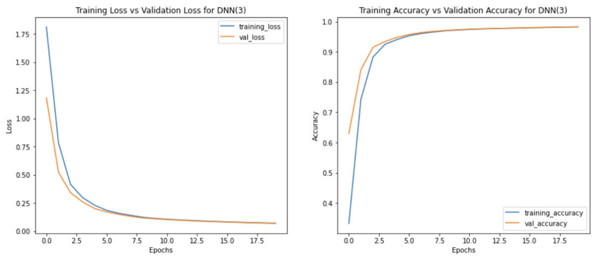


FIGURE 14. Training & validation performance DNN(3).

over time, the validation accuracy reaches a plateau after a few epochs. In this instance, the validation loss decreases steadily over time, indicating that the model is improving its ability to generalise to the validation set. This indicates that the model fits the training data and does not overfit less tightly than its antecedent. In conclusion, regarding training and validation, the figures illustrate the performance of the various levels of the DHMLM during their training phase, with the models capable of generalisation without overfitting. These numbers are useful for evaluating the effectiveness of different models and identifying problems such as overfitting and underfitting.

Table 6 displays DHMLM performance metrics by class. It contains four columns, namely Class, "Accuracy", "Precision", "Recall", and "F1", as well as three classification levels. Level 1 of the classification consists of two classes, namely, "Attack" and "Benign". The model has attained high precision for both classes, 99.61% for "Benign" and 99.28% for "Attack". The precision of "Attack" is marginally greater than that of "Benign", while their recall and F1 scores are nearly

TABLE 6. Class-wise performance metrics of DHMLM.

	Class	Accuracy	Precision	Recall	F1
Level 1	Benign	99.61	99.30	99.62	99.46
	Attack	99.29	99.61	99.30	99.45
Level 2	DDoS	99.71	99.78	99.71	99.74
	SQLi	99.62	99.68	99.62	99.65
	MITM	99.75	97.65	99.76	98.69
	Dictionary	97.70	99.73	97.70	98.71
Level 3	TFTP	98.35	97.10	98.35	97.72
	UDP+SSDP	98.41	99.67	98.41	99.04
	LDAP	99.52	98.21	99.52	98.87
	NetBIOS+Portmap	99.60	99.97	99.61	99.79
	SYN	99.77	98.71	99.77	99.24
	MSSQL	95.97	97.96	95.98	96.96
	HULK	98.90	98.39	98.91	98.65
	RUDY	99.41	99.45	99.42	99.44
	GoldenEye	98.27	98.81	98.28	98.54

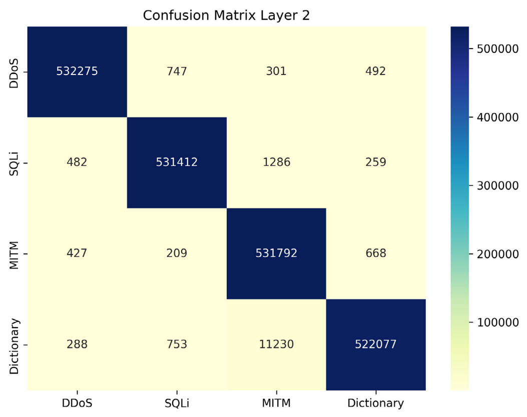


FIGURE 16. DNN(2) confusion matrix.

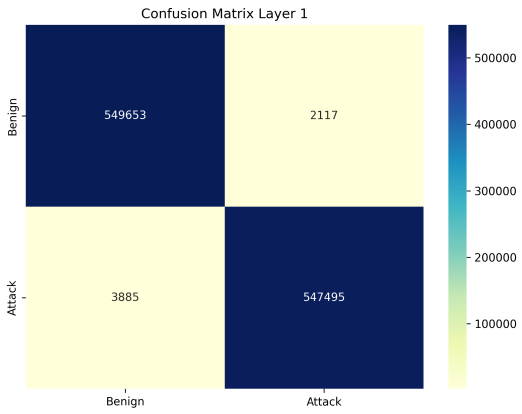


FIGURE 15. DNN(1) confusion matrix.

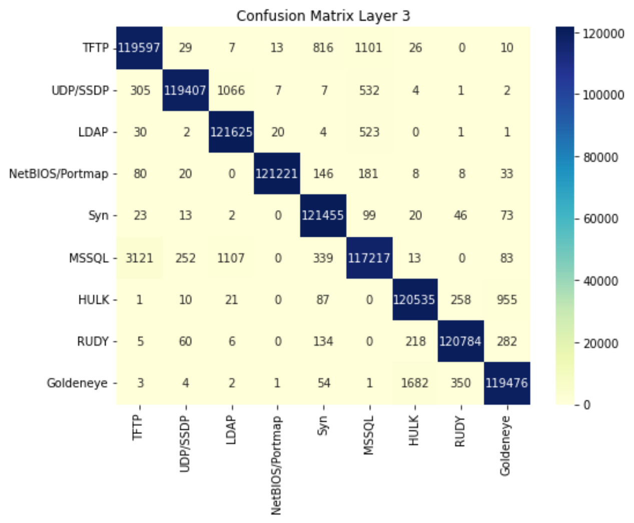


FIGURE 17. DNN(3) confusion matrix.

identical. Four classes comprise Level 2 of the classification: "DDoS", "SQLi", "MITM", and "Dictionary". The model's accuracy for each of the four divisions ranges from 97.70% to 99.75%. The precision, recall, and F1 scores range between 97.65% and 99.78% for all classes. Nine classes comprise Level 3 of the classification: "TFTP", "UDP+SSDP", "LDAP", "NetBIOS+Portmap", "SYN", "MSSQL", "HULK", "RUDY", and "GoldenEye". The model's accuracy for each of the nine classes ranges from 95.97% to 99.77%. The precision, recall, and F1 scores range from 95.98% to 99.97% across all classes.

The class-wise accuracy, precision, recall, and F1 score obtained by the three layers of the DHMLM on the test set is shown in Table 6. The confusion matrices generated by running the training and identification of the same data set are also shown below in Fig. 15, Fig. 16 and Fig. 17. DHMLM's accuracy, precision, recall, and F1 score performance are impressive overall. The model has obtained high performance for all classes at all three classification levels, indicating that it can effectively classify network traffic into various categories of attacks and benign traffic. This analysis can be used to enhance the security of computer networks by detecting and preventing network attacks more precisely.

The given Appendix Figures 26, 27, 28 and 29 represent the Receiver Operating Characteristic (ROC) curves for different Deep Neural Networks (DNNs). ROC curve plots the true positive rate (TPR) and false positive rate (FPR) for a binary classifier system, where TPR is plotted on the y-axis and FPR on the x-axis. The area under the ROC curve (AUC-ROC) measures the classifier's performance. A perfect classifier has an AUC-ROC of 1 or close to 1 (e.g. 0.99-1.00), while a random classifier has an AUC-ROC of 0.5.

Figure 26 shows the AUC-ROC curve for the first level of the Deep neural network of DHMLM (named DNN(1)), which appears to have an AUC-ROC of around 0.99 for the single class that level one identifies (it identifies if there is an attack or not), indicating a very high level of performance. Next, Figure 27 shows the AUC-ROC curves for the second level in the Deep Hierarchical Machine Learning Network (named DNN(2)) for the multiple class identification (for the attacks DoS-DDoS, SQLi, MiTM, and dictionary), which also appears to have an AUC-ROC of around 0.99, indicating a very high level of performance. Continuing, Figures 28 and 29 shows the AUC-ROC curves at the third

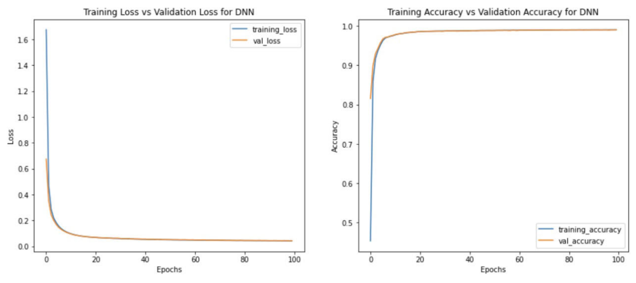


FIGURE 18. Training and validation performance DNN.

level in the Deep Hierarchical Machine Learning Network (named DNN(3)) for the multiple class identification (for the attacks TFTP, UDP+SSDP, NetBIOS+POrtmap, SYN, MSSQL, HUL, RUDY, and GoldenEye), which also appears to have an AUC-ROC of around 0.99, indicating a very high level of performance.

Regarding the AUC-ROC curves, all three figures have very similar shapes. The curves closest to the upper left corner have the best performance, and those closest to the diagonal line have the worst performance (the middle line represents the random approach). The highest AUC-ROC curve in each figure appears to be around 0.99, indicating excellent performance. More specifically, our model's Area Under Curve - Receiver Operator Characteristic (AUC-ROC) curves show that it is a nearly perfect classifier as the TPR for each class is extremely high compared to the FPR. The AUC-ROC curves for our model are shown in Fig. 26, Fig. 27, Fig. 28 and Fig. 29.

In conclusion, the Deep Hierarchical Machine Learning Model (DHMLM) is a highly efficient neural network model for classifying network traffic as benign or malicious. Due to its three-level classification system, it can identify attack traffic and classify it into specific attack categories and subcategories. The model's remarkable performance metrics, which include high accuracy, precision, recall, and F1 scores, demonstrate its ability to classify network traffic accurately. In general, the DHMLM is a useful instrument for network security professionals in identifying and mitigating cyber attacks.

b: RESULTS OF THE DNN

For DNN, Figure 18 depicts the performance of the DNN model during training and validation. The accuracy and loss values for training and validation converge as training continues, indicating that the model is correctly learning the data's characteristics. The efficacy of a Deep Neural Network (DNN) in categorising network traffic into distinct categories is displayed in Table 7. Accuracy, precision, recall, and the F1 score for each class are the performance metrics used to evaluate the model's performance. The model's aggregate accuracy for this task was 96.57%, which is acceptable. Examining the performance by category reveals that the DNN model performed well in classifying the vast majority of traffic categories. For the "Benign", "TFTP", "NetBIOS+Portmap", "SYN", "MITM", "RUDY" and "GoldenEye" classes, it attained high accuracy, precision, recall, and F1 score. For instance, the model's accuracy for these classes was 98.91%, 99.02%, 99.44%, 99.30%, 99.48%, 98.86%, and 97.54%, respectively. The model can precisely classify traffic in these categories based on the model's high accuracy and other performance metrics. However, the model performed poorly when classifying traffic in certain categories. For the "LDAP", "UDP+SSDP", "MSSQL", "SQLi" and "Dictionary" classes, for instance, it attained low accuracy, precision, recall, and F1 score. For instance, the model's accuracy for the "LDAP" class was only 1.99%, indicating that it performed very inadequately in this category. Similarly, the model's performance metrics for the "UDP+SSDP", "MSSQL", "SQLi" and "Dictionary" classes were poor. This suggests that the model requires further development to accurately classify traffic into these categories. Moreover, Figure 19 depicts the Confusion Matrix for the DNN model, which provides a more detailed view of each class's performance. The scatter plot demonstrates that the model performed well in classifying the "Benign" and "TFTP" classes but struggled with the "LDAP," "UDP+SSDP," "MSSQL," "SQLi," and "Dictionary" classes. Overall, the DNN model adequately classified network traffic into distinct categories. However, the model requires further development to classify traffic into certain categories accurately. The performance metrics and confusion matrix diagrams can be used to identify areas of the model that require development, which can then be utilised to create a more accurate model. In conclusion, the Deep Neural Network (DNN) model performed well in classifying network traffic into distinct categories, attaining an overall accuracy of 96.57%. However, the model's efficacy in classifying certain categories was inadequate, emphasising areas in need of further development. Utilising performance metrics and confusion matrix diagrams aided in the identification of these areas. The model has the potential to enhance its performance and become even more accurate at classifying network traffic with additional refinement.

TABLE 7. Class-wise performance metrics of DNN.

Class	Accuracy	Precision	Recall	F1
Benign	98.91	79.18	98.91	87.95
TFTP	99.02	97.96	99.02	98.49
UDP+SSDP	99.41	47.43	99.41	64.22
LDAP	1.99	46.58	1.99	3.81
NetBIOS+Portmap	99.44	99.85	99.44	99.64
SYN	99.30	98.66	99.30	98.98
MSSQL	85.71	98.99	85.71	91.87
HULK	98.65	80.37	98.65	88.58
RUDY	98.86	99.43	98.86	99.15
GoldenEye	97.54	98.85	97.54	98.19
SQLi	74.70	99.68	74.70	85.40
MITM	99.48	98.31	99.48	98.89
Dictionary	76.36	99.48	76.36	86.40

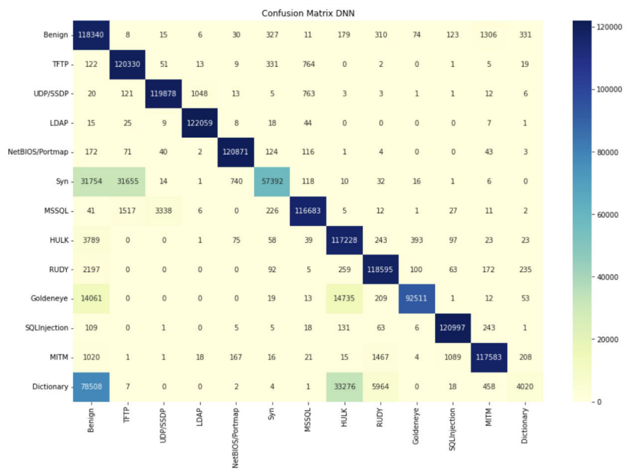


FIGURE 19. DNN confusion matrix.

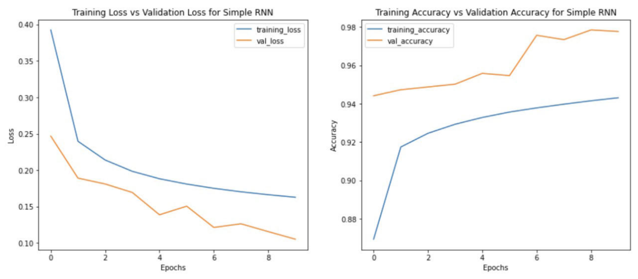


FIGURE 20. Training and validation performance RNN.

c: RESULTS OF THE RNN

For RNN, the training and validation performance of the RNN model is depicted in Figure 20. The training and validation loss decreases over successive epochs, indicating that the model is learning and not overfitting the data. The increasing training and validation accuracy over the epochs indicates that the model's performance on the data is improving. Table 8 displays the RNN model's class-specific performance metrics. The model obtained a high degree of accuracy for most categories, with an aggregate accuracy of 99.57%. However, the MSSQL class accuracy is considerably lower at 99.04%, indicating that the model has trouble identifying this class. For the majority of classes, the precision values are high, indicating that the model makes fewer false-positive predictions. Most class recall values are high, indicating that the model can identify the majority of instances of the class. The high F1 scores for most classes indicate a balance between precision and recall. Figure 21 depicts the RNN model's confusion matrix. The diagonal elements represent the number of correctly classified instances, while the off-diagonal elements represent the misclassified instances. The confusion matrix demonstrates that the model performs well for most classes, with a few misclassifications. However, the model has trouble identifying the MSSQL class, as evidenced by the lower number of true positives and higher number of false negatives for that class. Overall, the RNN model performs well for most categories, with an overall accuracy of 99.57%. However, the model needs help distinguishing the MSSQL class, and additional work is required to enhance the model's efficacy for this class. In conclusion, the RNN model has demonstrated promising classification accuracy for the majority of network traffic data classes. However, MSSQL class enhancements and additional research are required to enhance the model's efficacy. Overall, the model's performance demonstrates the potential of using RNNs to classify network traffic, and future research can build upon these findings to create more accurate and effective models.

TABLE 8. Class-wise performance metrics of RNN.

Class	Accuracy	Precision	Recall	F1
Benign	99.75	63.79	88.13	74.01
TFTP	99.98	93.24	98.71	95.89
UDP+SSDP	99.43	99.99	97.36	98.66
LDAP	99.81	98.15	99.89	99.01
NetBIOS+Portmap	99.91	99.98	99.62	99.80
SYN	99.83	99.07	99.38	99.22
MSSQL	99.04	06.01	90.65	11.28
HULK	99.52	99.13	97.74	98.43
RUDY	99.93	99.69	98.09	98.88
GoldenEye	99.57	89.26	94.86	91.97
SQLi	99.98	99.90	99.44	99.67
MITM	99.88	86.00	87.56	86.77
Dictionary	99.97	90.55	98.36	94.29

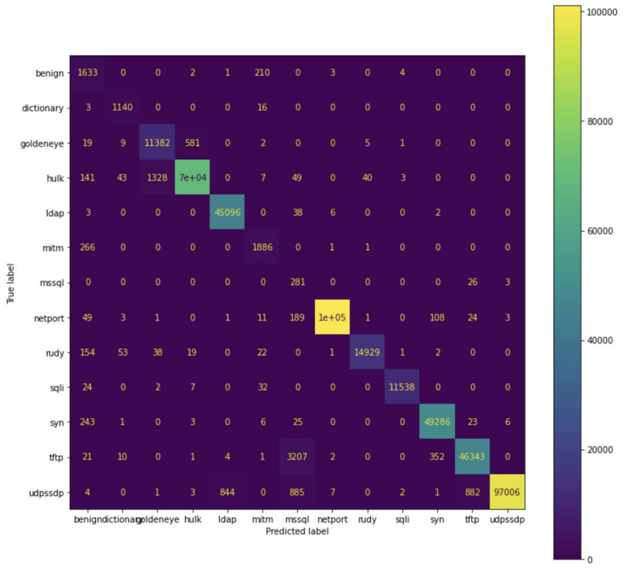


FIGURE 21. RNN confusion matrix.

d: RESULTS OF THE LSTM

For LSTM, the model's performance during training and validation is depicted in Figure 22. Over time, the training and validation loss decreases, while the validation accuracy remains high, indicating that the model generalises well to new data. Table 9 displays the class-wise performance metrics of an LSTM model trained with a particular dataset.

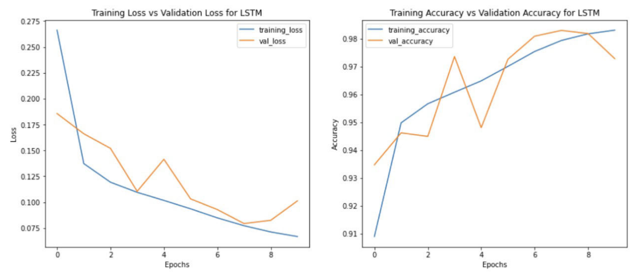


FIGURE 22. Training and validation performance LSTM.

TABLE 9. Class-wise performance metrics of LSTM.

Class	Accuracy	Precision	Recall	F1
Benign	99.93	85.99	99.03	92.05
TFTP	99.39	98.89	95.422	97.12
UDP+SSDP	99.62	99.85	98.39	99.12
LDAP	99.80	98.11	99.92	99.00
NetBIOS+Portmap	99.91	99.98	99.62	99.80
SYN	99.86	99.06	99.68	99.37
MSSQL	99.51	11.68	95.16	20.81
HULK	99.52	99.76	97.15	98.44
RUDY	99.93	98.62	99.29	98.95
GoldenEye	99.57	86.79	98.36	92.21
SQLi	99.99	99.93	99.60	99.77
MITM	99.97	95.61	98.14	96.86
Dictionary	99.98	93.24	98.71	95.89

The model's accuracy ranges from 99.39% to 99.99%, indicating that it detects various types of network traffic with high precision. Precision measures the model's ability to distinguish between genuine positives and false positives. The precision values range from 11.68% (for MSSQL) to 99.98% (for Dictionary), indicating that the model accurately identifies positive samples for the majority of classes, with the exception of MSSQL. The recall metric measures the model's ability to correctly identify all affirmative samples, with values ranging from 95.16 to 99.92% (for MSSQL and LDAP, respectively). The high recall values indicate that the model is capable of identifying the majority of positive samples for all classes. The F1 score, which is the harmonic mean of precision and recall, represents the model's performance as a whole. F1 values range between 20.81% (MSSQL) and 99.77% (SQLi). The low F1 value for MSSQL suggests that the model needs to identify this traffic class. The LSTM model's confusion matrix is depicted in Figure 23. The diagonal values represent the number of properly classified samples for each class, whereas the off-diagonal values represent misclassifications. The confusion matrix indicates that the model has difficulty correctly identifying MSSQL class samples, which is consistent with the class's low precision and F1 score. Except for MSSQL, the LSTM model appears to function well in detecting various types of network traffic. In conclusion, the LSTM model demonstrates a high degree of accuracy, precision, recall, and F1 score when identifying different categories of network traffic. While there is room for development in identifying MSSQL traffic, the model's generalisation ability suggests it could be a useful classification tool for network traffic. Future work could concentrate on enhancing the model's performance for MSSQL and possibly investigating other models or techniques to further improve the classification accuracy of network traffic.

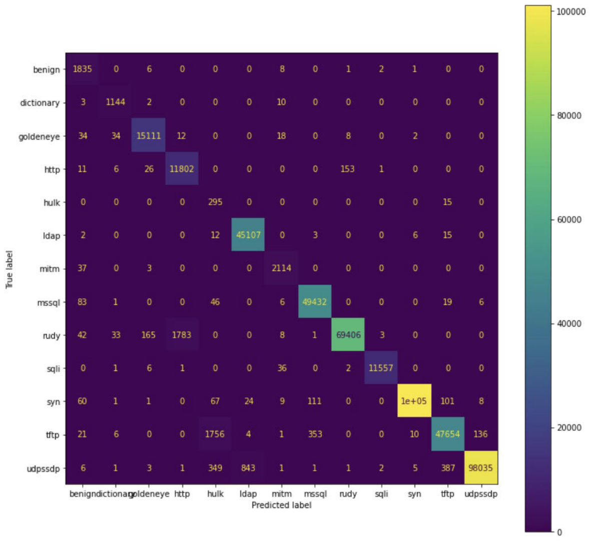


FIGURE 23. LSTM confusion matrix.

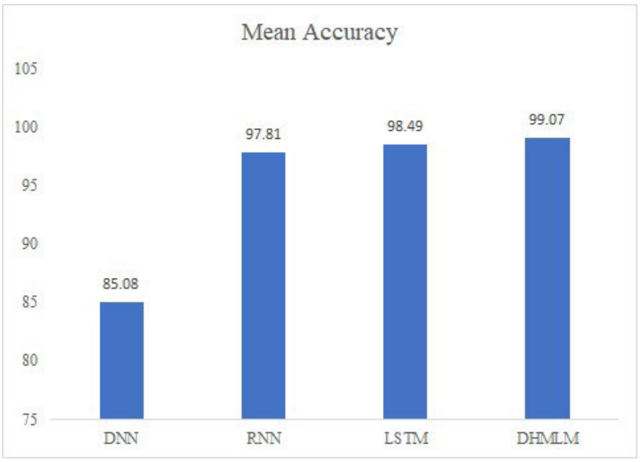


FIGURE 24. Mean accuracies of investigated approaches.

2) COMPARISON AMONG DIFFERENT MODELS LIKE DHMLM, DNN, RNN, AND LSTM

The following section provides a comparison among the investigated approaches based on the resulting confusion matrices shown in Figures 15, 19, 21 and 23 and the resulting training and validation loss shown in Figures 12, 18, 20 and 22 along with the findings in Figure 24. So, the overall DNN accuracy of 86.84% on the data set for a 13-class classification using 972 connections is far inferior to the 747 connections and 99.07% accuracy of the DHMLM. The DNN's classwise metrics are illustrated in Table 7. Table 10 shows the accuracy confusion matrix metric with other metrics such

TABLE 10. Comparison of DHMLM, DNN, RNN, and LSTM.

Model	Accuracy	Epochs	Training Time(s)	Neurons	Connections	Prediction Time(μs)
DNN	86.43%	100	77.7	60	972	24.6
DHMLM	99.07%	60	43.77	47	747	16.3
RNN	97.81%	10	1753	50(RNN cells)	160	28.00
LSTM	98.49%	10	2097	50(LSTM cells)	160	49.83

as epochs used to train the model, training time, neurons, connections, and prediction time. The multi-class classification accuracy of the RNN was 97.81% with 50 RNN cells and 160 connections only. The LSTM outperformed the RNN in terms of accuracy with similar parameters as expected. The mean accuracy of the LSTM was 98.49%. It also had the highest training and inference times of 2097 seconds and 49.83 (μs), respectively, despite having the least number of connections. The RNN and LSTM had significantly higher training and prediction times than the DNN and the DHMLM. As shown in Table 10, the training time of the DNN was 77.7 seconds in contrast to the DHMLM's 43.77 seconds which is only 56% the time taken by the DNN. Moreover, the training time of the RNN was 1753 seconds and the LSTM was 2097 seconds which are significantly higher than the aforementioned approaches. So, the low train time of the DHMLM has some positive implications, the most important among them being the increased real-world effectiveness of the DHMLM as the turnaround time is much lesser on retraining and the computational intensity is low. In the following paragraph, we examine in terms of attacks how each approach performance achieves (according to the Tables 7, 8 and 9):

- The RNN outperformed the DNN with an accuracy of at least 99% in every single class. It showed the least accuracy for MSSQL class which has the least number of data points. This is consistent with the DHMLM having the lowest accuracy for the same class. On the other hand, the TFTP and SQLi attacks were predicted with a high accuracy of 99.98%. Although the DHMLM did not show 99% and above accuracy for every single class, the precision, recall, and F1 scores were considerably higher. The RNN yielded a poor precision of 06.01% for the MSSQL class.
- The LSTM exhibited similarly high accuracies with RNN; however, it outperformed the RNN regarding precision, recall, and F1 scores. The worst precision value still belonged to the MSSQL class at 11.68%. Both the RNN and LSTM had the lowest F1 score for MSSQL class.
- The DNN performs the best in classifying MITM attacks with an accuracy of 99.48%, and it performs almost equally well on TFTP, UDP+SSDP, and NetBIOS+Portmap. On the other hand, the LDAP attack had the least accuracy, with a meager 1.99%. Further, the DNN performed poorly in classifying Dictionary and SQLi attacks, and the rest of the attacks related to the achievements of the other approaches.
- The DHMLM outperforms the DNN in classifying all classes other than TFTP and UDP+SSDP in terms of accuracy, precision, recall, and F1 score, including benign packet classification with an accuracy of 99.61% compared to the DNN's 98.91 %. Our model identifies the SYN attack with a maximum accuracy of 99.77% and performs similarly well on "SQLi", "LDAP", "NetBIOS+Portmap", "MITM", and "RUDY". The model also performs well on attacks like LDAP where the DNN showed poor results. The RNN, LSTM, or DNN did not outperform the DHMLM at the precision, recall, and F1 scores. The aforementioned values were not consistently higher than at least 96%. Moreover, as shown in Section VI-B5, the DNN, LSTM, and DNN cannot act as a Zero-day attack classifier as opposed to the DHMLM, whose first level performed the task with 99.63 % accuracy.

Finally, Figure 24 shows that the DHMLM is the most accurate approach.

3) COMPARISON OF COMPUTATIONAL COMPLEXITY DIFFERENCES BETWEEN THE INVESTIGATED MODELS OF DNN, RNN, LSTM MODELS AND THE FINAL PROPOSED MODEL OF DHMLM

This section compares the Big-O notation among the papers' investigated models. More specifically, when building and utilising algorithms, a typical investigation is whether an algorithm is more effective in completing a particular task. One method that can be used for this investigation is the Big-O notation analytical method. The Big-O notation is a method for analysing the complexity of an algorithm. Big-O is concerned with the asymptotic time complexity of an algorithm or the upper bound on the number of steps required relative to the amount of data being processed [78]. Few efforts have been made in the literature to exploit the hierarchical nature of categories. Nevertheless, hierarchical models perform better than flat models in various areas for picture categorisation [11], [79], [80]. Because Hierarchical decomposition simplifies complex issues by breaking them down into smaller, interconnected issues. The smaller problems are solved independently, and then the answers are recombined to solve the original problem. Consequently, one of the primary objectives of hierarchical learning is the reduction of computing complexity. Using a collection of multi-level classifiers can lower the cost of learning, according to the proposed approach. Undoubtedly, the collection of multi-level classifiers constitutes a hierarchical learning structure [11], [79], [80].

Considering computational complexity according to [81], the computational complexity of a feed-forward neural network was examined, as well as why it is advantageous to separate the computation into training and an inference phase as back-propagation (n^5) is substantially slower than forward propagation (n^4). Thus, in our case, the models used are all deep neural networks that work with back-propagation; hence, they all have the same Big-O computational complexity of n^5. But, empirically, as aforementioned, with the observation DHMLM has reduced execution time and is less than other models, which can be verified from Section VI-B4.

4) COMPARATIVE REMARKS OF COMPETITIVE APPROACHES FOUND IN OPEN LITERATURE USING THE UNIFIED DATASET

As shown in Table 11, the proposed DHMLM model outperforms Neural Networks Multi-Classifier [65], OKNN [73], and EagerNet [74] in every metric while being trained for a lesser number of epochs and having a significantly shorter training time. Also, the DHMLM comprises only 47 neurons, a small number compared to Neural Networks Multi-Classifier [65], which is implemented with 896 neurons, and EagerNet's [74], which is implemented with 640 neurons. Note that OKNN[9] is not implemented with neurons. Thus, the number of neuron connections in our proposed model is correspondingly notably lower. Moreover, in terms of complexity and training cycle, the DHMLM obtains the lowest complexity by completing a training cycle of 60 epochs in 43.77 seconds, followed by the OKNN [73], which has no epoch cycles due to its implementation and training time of 669 seconds. Continuing, the Neural Networks Multi-Classifier [65] comes in third place with a training cycle of 100 epochs and a training time of 786 seconds, followed by the EagerNet's model with a training cycle of 800 epochs and a training time of 969 seconds in the last place. Moreover, the investigated approach achieves the best execution/inference time with a value of 16.3 (μs), followed by the OKNN [73] with a value of 19.11 (μs), Networks Multi-Classifier [65] with a value of 32.17 (μs) and last, is the EagerNet [74] with a value of 52.13 (μs). Therefore, as shown, the proposed approach has better performance in terms of training, execution time, and complexity (based on the neurons and epochs needed for training) and ultimately better results in terms of Accuracy, Recall, Precision, and F1 Score than the other approaches examined in this section with the use of our unified dataset.

5) ZERO-DAY/UNKNOWN ATTACKS IDENTIFICATION BY OUR SELECTED APPROACH OF DHMLM

The emulated HTTP Flood attack, TCP Flood attack, UDP Flood attack, and MITM REST attack are the Zero-days and unknown attacks to test our model. These attacks were omitted from our training data set and thus are completely new to our model. The data set used and the results of the experiment are shown below in Table 12.

As seen above, the first layer of our model is able to identify the HTTP Flood, TCP Flood, UDP Flood, and MITM Representational State Transfer Application Programming Interface (REST API) Zero-day/unknown attacks with an

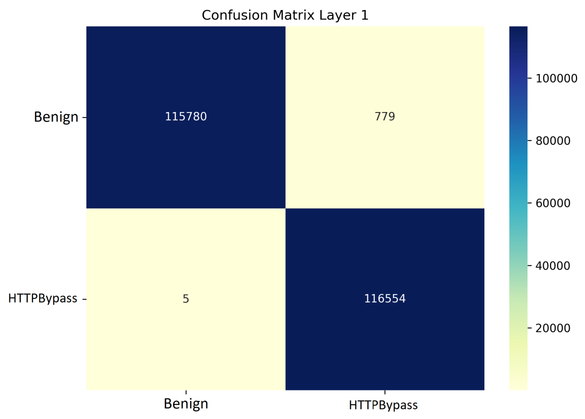


FIGURE 25. DNN(1) confusion matrix for zero day attack.

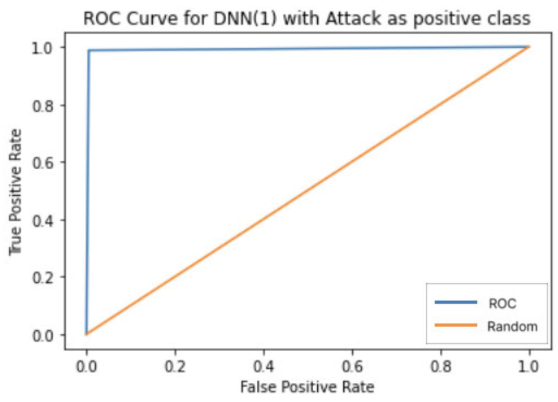


FIGURE 26. AUC-ROC curve for DNN(1) - Attack?

accuracy of 99.63%, 99.62%, 99.61%, and 99.96% respectively. Also, it can identify the Note that the analysis of an unknown attack is straightforward, and we are currently investigating whether or not the unknown attack data belongs to the attack type. In the future, we will investigate the possibility of confirming whether an attack does not correspond to a known attack type.

DHMLM can identify unknown attacks due to its working mechanism; it discovers a mapping from a low-level feature space to a high-level conceptual space or from an adequate space to its quotient space (a coarse space). Taking into account that certain attacks share common characteristics and that these characteristics are represented in the PCAP file analysis as specific features from the selected features, they are represented in the model as low-level features that activate the activation function (that decides whether the neuron should be activated or not) resulting in weights being matched with known and unknown similar attacks (along with the bias). For instance, HTTP Flood is a comparable DDoS attack (in which our model is trained). This is an admirable quality of DHMLM.

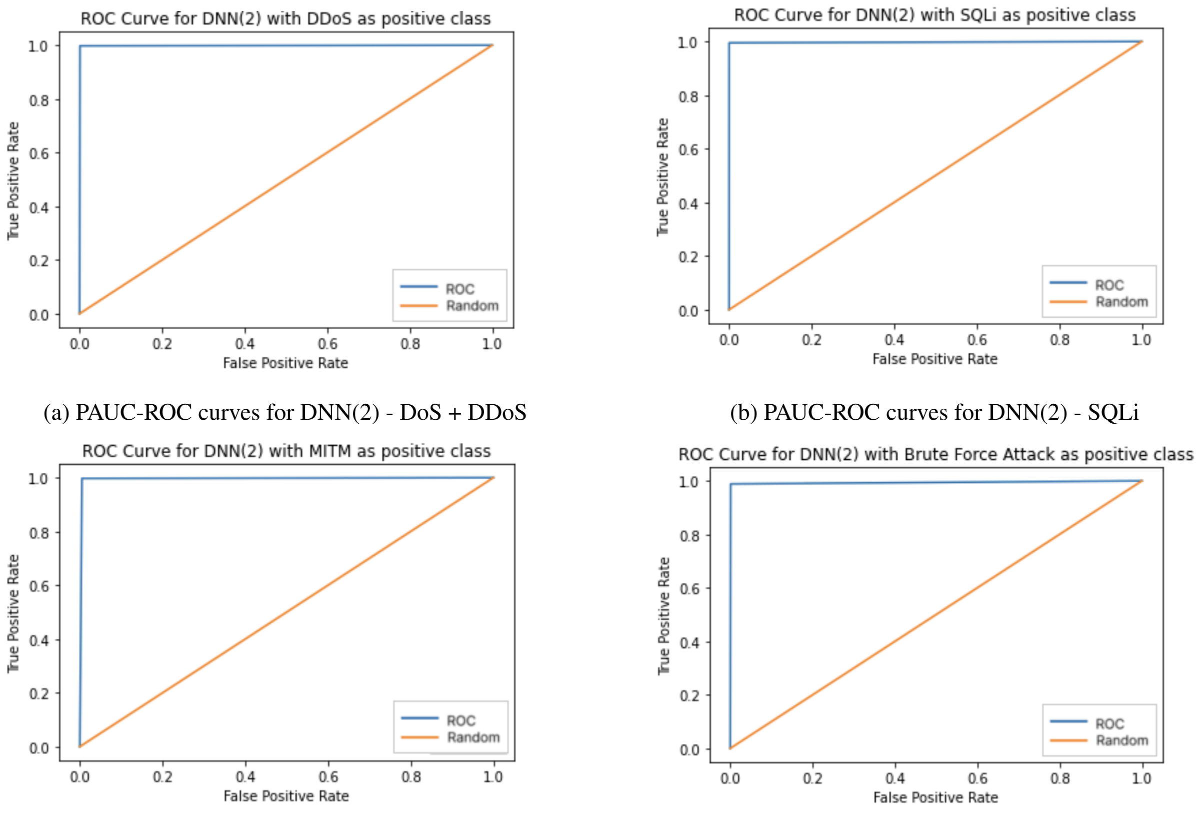
6) OVERALL REMARKS

We have seen the performance of the DHMLM compared to some of the proposed paper models and then with models proposed by recent papers. Firstly, we compared its performance

[9]Because OKNN is a K-Means base algorithm is an approach without neurons (that is why K-Means base algorithm is an unsupervised learning technique, while the other methods are supervised learning techniques; thus, the training time of 669 s represents the identification of the best k number that represents the number of clusters and the separation of the points to clusters using the nearest neighbours along with the training, validation, and testing.

TABLE 11. Comparative study of recent papers with the Unified Dataset.

Approach	Neural Networks Multi-Classifier[67]	OKNN[75]	EagerNet[76]	DHMLM
Accuracy	94.21	92,19	99,05	99.07
Recall	94.03	94.60	92,85	98.95
Precision	94.21	89.80	91,15	98.91
F1 Score	94.12	82.60	91,03	98.93
Epochs	100	N/A	800	60
Training Time(s)	786	669	969	43.77
Execution / Inference Time (μs)	32.17	19.11	52.13	16.3
Neurons	896	N/A	640	47



(a) PAUC-ROC curves for DNN(2) - DoS + DDoS

(b) PAUC-ROC curves for DNN(2) - SQLi

(c) PAUC-ROC curves for DNN(2) - MiTM

(d) AUC-ROC curves for DNN(2) - Brute Force Attack

FIGURE 27. AUC-ROC curves for DNN(2).

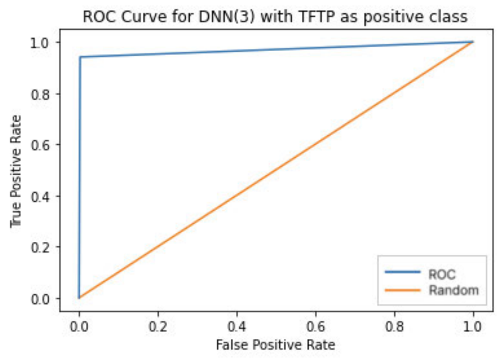
TABLE 12. Zero-day attack data set.

Class	Number of Data points
Benign	116,599
HTTP Flood	116,599
TCP Flood	116,599
UDP Flood	116,599
MITM REST	116,599
Total	466,396

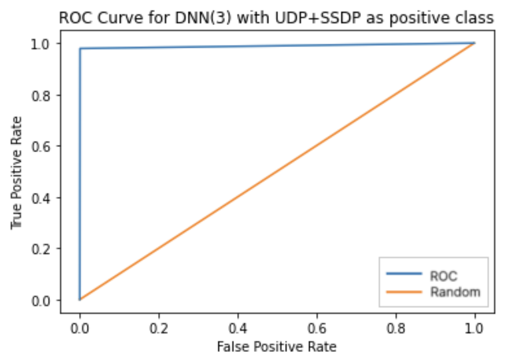
TABLE 13. Performance metrics on zero-day attack data set.

Class	Accuracy	Precision	Recall	F1
Benign	99.63	99.33	99.99	99.65
HTTP Flood	99.63	99.99	99.33	99.65
TCP Flood	99.62	99.99	99.23	99.60
UDP Flood	99.61	99.99	99.12	99.55
MITM REST	99.96	95.61	98.14	96.86

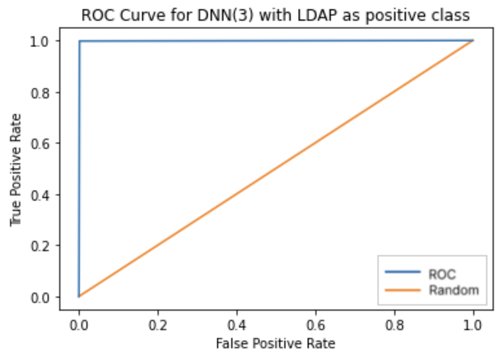
to a DNN, RNN and LSTM and observed that the hierarchical design helped achieve a much higher accuracy of 99.07% compared to the LSTM with 98.49%, RNN with 97.81% and lastly, DNN with 86.84%. Also, the training time of the proposed model has shown a 46%, 97.5%, and 98% improvement compared to DNN, RNN, and LSTM, respectively. While at the same time, using 13, 3, and 3 neurons compared to DNN, RNN, and LSTM, respectively, and lastly, 40 fewer epochs on the training compared to DNN. However, for RNN and LSTM, DHMLM has 50 more epochs. The proposed model also has 225 connections, thus having a lower complexity than the DNN, but 587 more than RNN and LSTM. Regarding execution time, the DHMLM with 16.3 μs has the best time



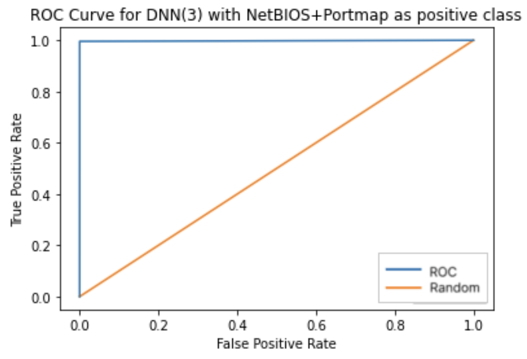
(a) AUC-ROC curves for DNN(3) - TFTP



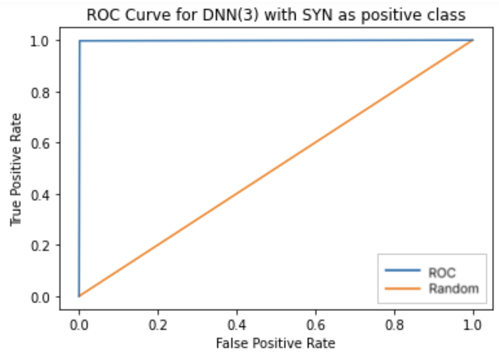
(b) AUC-ROC curves for DNN(3) - UDP + SSDP



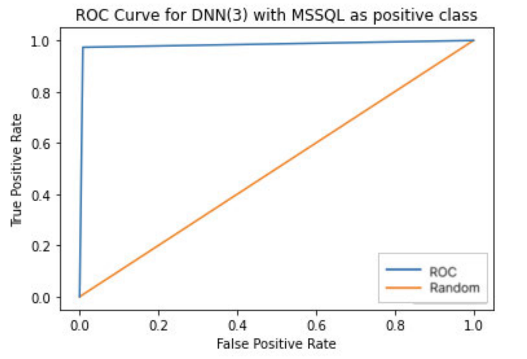
(c) AUC-ROC curves for DNN(3) - LDAP



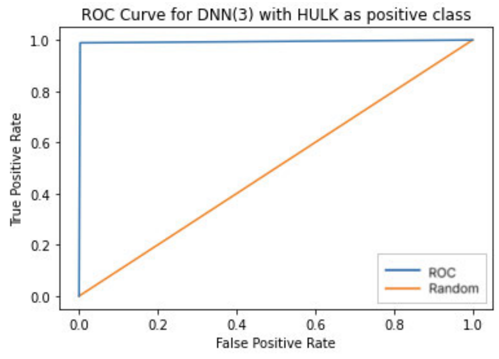
(d) AUC-ROC curves for DNN(3) - NetBIOS and Portmap



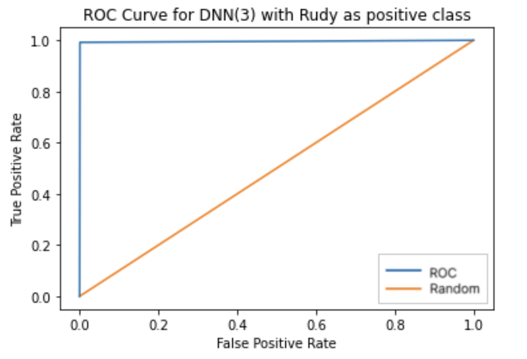
(e) AUC-ROC curves for DNN(3) - SYN



(f) AUC-ROC curves for DNN(3) - MSSQL



(g) AUC-ROC curves for DNN(3) - H.U.L.K.



(h) AUC-ROC curves for DNN(3) - R.U.D.Y.

FIGURE 28. AUC-ROC curves for DNN(3).

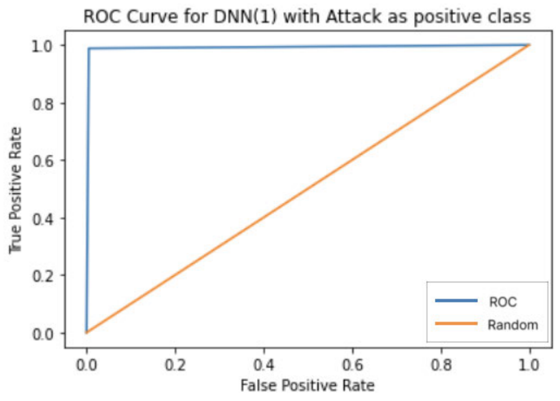


FIGURE 29. AUC-ROC curves for DNN(3) - Goldeneye.

compared to the DNN with 24.6 μs, RNN with 28 μs, and LSTM with 49.83 μs. Secondly, from the open literature comparison, the DHMLM achieves an overall accuracy of 99.07%, which is a substantial increase on Neural Networks Multi-Classifier [65] (94.21%), OKNN [73] (92.19%) and EagerNet [74] (99.05%). The proposed model also has better recall, precision, and accuracy scores of 98.95%, 98.91%, and 98.93% when compared to OKNN's [73] (94.6%, 89.8%, and 82.6%), to Neural Networks Multi-Classifier [65] (94.03%, 94.21%, and 94.12%) and EagerNet [74] (92.85%, 91.15%, and 91.03%). Continuing, the DHMLM's 60 epoch training cycle and usage of 47 neurons are significantly lesser than EagerNet's [74] 800 epoch cycle and 640 neuron usage and Neural Networks Multi-Classifier [65] 100 epoch cycle and 896 neuron usage. In terms of training and execution time, the DHMLM (with 43.77 & 16.3 μs) has the best times compared to the Neural Networks Multi-Classifier [65] (with 786 & 32.17 μs), the OKNN [73] (with 669 & 19.11 μs) and the EagerNet [74] (with 969 & 52.13 μs). Finally, we tested the first layer of the DHMLM against Zero-day/unknown attacks along with benign packets, which the model could recognise and classify with 99.63% accuracy. Overall the DHMLM has shown better performance over all the investigated approaches and the major approaches from the literature review.

VII. CONCLUSION AND FUTURE WORK

D2D network security solutions like IDS trained with AI/ML are absolutely necessary due to the inherent vulnerability and lack of standardisation of security measures. Thus, our work focuses primarily on the identification of the most significant threats to D2D networks, namely DDoS (i.e., TFTP, UDP+SSDP, LDAP, NetBIOS+Portmap, SYN, MSSQL, HULK, RUDY, GoldenEye), SQLi, MITM and Dictionary attacks. This research is important because it can identify all the threads mentioned above and classify them with the least time for execution and training. Additionally, the need for a robust and real-time IDS necessitates adopting a computationally lighter DHMLM that operates on the data set's hierarchical structure that can run on mobile devices (as shown in Section VI-B3). The proposed system demonstrates applicability to the base stations, which serve as the gateways for the D2D communication protocol, as per the given context (as shown in [2], [59]). Additionally, the proposed system is applicable to the D2D-Relay and D2D Multi-Hop Relay (D2DMH-Relay) devices if the device models are of the latest technology (as shown in [2], [59]).

Regarding features selection and model training, we selected the features used for our model training using a

TABLE 14. Abbreviation table.

AAHE	Attribute-based Adaptive Homomorphic Encryption
ANNs	Artificial Neural Networks
ANN	Artificial Neural Network
Ada Boost	Adaptive Boost
AP	Access Point
AR	Attribute Ratio
ART	Adaptive Resonance Theory
AUC-ROC	Area Under Curve - Receiver Operator Characteristic
Ada Boost	Adaptive Boost
BCE	Binary Cross Entropy
BIOS	Basic Input/Output System
BP	Back Propagation
BPTT	Backpropagation Through Time
BS	Base Station
BWO	Black Widow Optimisation
CAIDA	Cooperative Association for Internet Data Analysis
CCE	Categorical Cross Entropy
CDR	Call Detail Record
CECs	Constant Error Carousels
CECs	constant error carousels
CFA	Cuttlefish Algorithm
CIC-DDoS2019	
	Intrusion Detection Evaluation Dataset CIC-DDoS2019 Evaluation Dataset
CIC FlowMeter	Flow meter
CLDAP	connectionless Lightweight Directory Access Protocol (LDAP)
CNN	Convolutional Neural Network
CSV	comma-separated values
D2D	Device-to-Device

TABLE 14. *(Continued.)* **Abbreviation table.**

D2D-Relay	D2D Relay device
D2DMH-Relay	D2D Multi-Hop Relay device
DDoS	Distributed denial of service
DDQN	Double Deep Q-learning Network
DHCP	Dynamic Host Configuration Protocol
DHML	Deep Hierarchical Machine Learning
DHMLM	Deep Hierarchical Machine Learning Model
DHNN	Deep Hierarchical Neural Network
DRL	Deep Reinforcement Learning
DNN	Deep Neural Network
DNNs	Deep Neural Networks
DNS	Domain Namne Server
DRC	Deep Rudimentary CNN
DRL	Deep Reinforcement Learning
DoS	Denial of Service
ETTERCAP	Ethernet Capture emulation for doing MiTM
FGLCC	Feature Grouping based on Linear Correlation Coefficient
FGMI	Feature Grouping based on Mutual Information
FL	Federated Learning
FPR	False Positive Rate
GTCS	Game Theory and Cyber Security
GUI	Graphical User Interface
HNN	Hierarchical Neural Network
HOIC	High Orbit Ion Canon
HTTP	Hypertext Transfer Protocol
HULK	HTTP-Unbearable-Load-King
ICMP	Internet Control Message Protocol
IDS	intrusion detection systems
IDSs	Intrusion Detection Systems
IT	Information Technology
IP	Internet Protocol
KDD	Knowledge Discovery and Data Mining
LANs	Local Area Networks
LCFS	Label Construction for Feature Selection
LDAP	Lightweight Directory Access Protocol
LGBM	light gradient-boosting machine
LOIC	Low Orbit Ion Cannon
LSTM	Long short-term memory
M-SMKL	interclass mean-squared difference growth/squared difference growth
MC-SQLR	Microsoft SQL Server Resolution Protocol
MI	Mutual Information-based
MIFS	Mutual Information-Based Feature Selection
MITM/ MiTM	Man in the Middle attack
ML	machine learning
ML-IDS	multi-level intrusion detection system
MLP	Multi-Layer Perceptron
MSSQL	Microsoft SQL Server
MYSQL/ MySQL	My-SQL
NN	Neural Networks
NTP	Network Time Protocol
NetBios	Network Basic Input/Output System
OBWO	Oppositional Based Black Widow Optimisation
OKNN	Optimum K-Nearest Neighbour
ONC	Open Network Computing
PCAP	Packet Capture
PHIDS	Parallel Hierarchical IDS
Portmap	Port Mapper
R.U.D.Y./RUDY	R U Dead yet
REST	Representational State Transfer
REST API	Representational State Transfer Application Programming Interface
RNN	Recurrent Neural Network
ROC	Receiver Operating Characteristic
RPC	Remote Procedure Call
RPCI	Remote Procedure Call Interface
ReLU	Rectified Linear Unit
S-SMKL	intra-class variance descent
SHIDS	Serial Hierarchical IDS
M-SMKL	interclass mean-squared difference growth/squared difference growth
SMOTE	Synthetic Minority Oversampling Technique
SMTP	Simple Mail Transfer Protocol

TABLE 14. ***(Continued.)*** **Abbreviation table.**

SNMP	Simple Network Management Protocol
SQL	Structured Query Language
SQLi	SQL Injection
SQLi	SQL Injection
SSDP	Simple Service Discovery Protocol
SYN	Synchronisation packets sends only in initial connection
TCP	Transmission Control Protocol
TFTP	Trivial File Transfer Protocol
TNN	Traditional Neural Network
TPR	True Positive Rate
UDP	User datagram protocol
UE	User Equipment
UPnP	Universal Plug and Play
WAP	wireless access point
XG Boost	Extreme Gradient Boosting

two-step process consisting of a correlation-based filter followed by a Mutual Information-based filter. A subset of 20 features produced the highest accuracy of 99.07% for the DHMLM. In addition, we have compared and contrasted the DHMLM with traditional DNN, RNN and LSTM, which are common classification solutions for attacks. The results demonstrated that the DHMLM is more suitable for low-powered devices because it achieves greater accuracy with fewer connections and better overall average performance while being trained for a substantially smaller number of epochs. The first level of the DHMLM can also identify an unknown attack with high accuracy (99.63% without retraining). The DHMLM outperforms the DNN, RNN, and LSTM regarding the retraining time when a new attack must be added to the classification task because it requires resetting fewer connections. Moreover, the training time can be reduced by 56%. In terms of Zero-Day/Unknown attacks, although the first level of our model can identify them, the second and third levels will be unable to classify them due to the absence of output nodes unique to these novel attacks. Thus, additional research can be conducted for future work on how Neural Networks can reorganise themselves and accurately identify Zero-Day attacks. In addition, the high correlation of 0.46 [65] between the NetBIOS and Portmap attacks and 0.68 [65] between the UDP and SSDP attacks makes classifying these attacks extremely difficult for ML models, with the majority of models achieving very low accuracy. This issue can also be investigated further. In addition, more work can be done on IDS at edge IoT networks, as we have currently focused only on IDS at D2D networks. Finally, other attacks besides the aforementioned examined attacks could be added to the data set to increase the model's sensitivity to variations in attack patterns. Moreover, as future work, we are planning to install our identification models shown in this paper and [56], at snort IDS and IPS (shown in [82]). Through Snort IPS/IDS, future research will show that it can block all the known and unknown attacks targeting the D2D and IoT communication networks from inside and outside of the network.

APPENDIX. AUC-ROC CURVES ORDER BY LEVEL AND ATTACK IDENTIFICATION OF THE DHMLM APPROACH

The given figures 26, 27, 28 and 29 represent the Receiver Operating Characteristic (ROC) curves for different Deep Neural Networks (DNNs). ROC curve plots the true positive rate (TPR) and false positive rate (FPR) for a binary classifier system, where TPR is plotted on the y-axis and FPR on the x-axis. The area under the ROC curve (AUC-ROC) measures the classifier's performance. A perfect classifier has an AUC-ROC of 1 or close to 1 (e.g. 0.99-1.00), while a random classifier has an AUC-ROC of 0.5.

Figure 26 shows the AUC-ROC curve for the first level of the Deep neural network of DHMLM (named DNN(1)), which appears to have an AUC-ROC of around 0.99 for the single class that level one identifies (it identifies if there is an attack or not), indicating a very high level of performance. Next, Figure 27 shows the AUC-ROC curves for the second level in the Deep Hierarchical Machine Learning Network (named DNN(2)) for the multiple class identification (for the attacks DoS-DDoS, SQLi, MiTM, and dictionary), which also appears to have an AUC-ROC of around 0.99, indicating a very high level of performance. Continuing, Figures 28 and 29 shows the AUC-ROC curves at the third level in the Deep Hierarchical Machine Learning Network (named DNN(3)) for the multiple class identification (for the attacks TFTP, UDP+SSDP, NetBIOS+POrtmap, SYN, MSSQL, HUL, RUDY, and GoldenEye), which also appears to have an AUC-ROC of around 0.99, indicating a very high level of performance.

Overall, the AUC-ROC curves in all three figures have very similar shapes. The curves closest to the upper left corner have the best performance, and those closest to the diagonal line have the worst performance (the middle line represents the random approach). The highest AUC-ROC curve in each figure appears to be around 0.99, indicating excellent performance.

APPENDIX. GLOSSARY OF TERMS AND ABBREVIATIONS

In this section, under the appendix, we provide the table of abbreviations used in the paper. Table 14 provides the abbreviation and description near it.

ACKNOWLEDGMENT

This research is part of a project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement N°739578 and the government of the Republic of Cyprus through the Directorate General for European Programmes, Coordination and Development. This work has received funding from the European Union under grant agreement N° 101087257. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union. Neither the European Union nor the granting authority can be held responsible for them. This work has received funding from the European Union under grant agreement N° 101095363. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union. Neither the European Union nor the granting authority can be held responsible for them.

REFERENCES

[1] R. I. Ansari, C. Chrysostomou, S. A. Hassan, M. Guizani, S. Mumtaz, J. Rodriguez, and J. J. P. C. Rodrigues, "5G D2D networks: Techniques, challenges, and future prospects," *IEEE Syst. J.*, vol. 12, no. 4, pp. 3970–3984, Dec. 2018.

[2] I. Ioannou, V. Vassiliou, C. Christophorou, and A. Pitsillides, "Distributed artificial intelligence solution for D2D communication in 5G networks," *IEEE Syst. J.*, vol. 14, no. 3, pp. 4232–4241, Sep. 2020.

[3] O. N. Hamoud, T. Kenaza, and Y. Challal, "Security in device-to-device communications: A survey," *IET Netw.*, vol. 7, no. 1, pp. 14–22, Jan. 2017.

[4] J. Sen and S. Mehtab, "Machine learning applications in misuse and anomaly detection," in *Security and Privacy From a Legal, Ethical, and Technical Perspective*. London, U.K.: IntechOpen, 2020, p. 155.

[5] A. L. Buczak and E. Guven, "A survey of data mining and machine learning methods for cyber security intrusion detection," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 2, pp. 1153–1176, 2nd Quart., 2016.

[6] N. Arica and F. T. Yarman-Vural, "An overview of character recognition focused on off-line handwriting," *IEEE Trans. Syst. Man Cybern. C, Appl. Rev.*, vol. 31, no. 2, pp. 216–233, May 2001.

[7] Z. Cao, J. Lu, S. Cui, and C. Zhang, "Zero-shot handwritten Chinese character recognition with hierarchical decomposition embedding," *Pattern Recognit.*, vol. 107, Nov. 2020, Art. no. 107488.

[8] X. Zheng, S. B. H. Shah, A. K. Bashir, R. Nawaz, and U. Rana, "Distributed hierarchical deep optimization for federated learning in mobile edge computing," *Comput. Commun.*, vol. 194, pp. 321–328, Oct. 2022.

[9] Y. Zhang and L. Han, "Learning (from) deep hierarchical structure among features," in *Proc. 33rd AAAI Conf. Artif. Intell., AAAI, 31st Innov. Appl. Artif. Intell. Conf., IAAI 9th AAAI Symp. Educ. Adv. Artif. Intell. (EAAI)*, 2019, pp. 5837–5844.

[10] M. Kearns, "The computational complexity of machine learning," M.S. thesis, MIT Press, Cambridge, MA, USA, 1990, p. 165.

[11] A. Barreto, B. V. Roy, Z. Wen, D. Precup, M. Ibrahimi, A. Barreto, B. Van Roy, and S. Singh, "On efficiency in hierarchical reinforcement learning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 6708–6718.

[12] L. Zhang and B. Zhang, "Hierarchical machine learning—A learning methodology inspired by human intelligence," in *Rough Sets and Knowledge Technology*, G.-Y. Wang, J. F. Peters, A. Skowron, and Y. Yao, Eds. Berlin, Germany: Springer, 2006, pp. 28–30.

[13] F. Gökgöz and F. Filiz, "Electricity price forecasting: A comparative analysis with Shallow-ANN and DNN," *Int. J. Energy Power Eng.*, vol. 12, no. 6, pp. 421–425, 2018.

[14] F. Anselmi, J. Z. Leibo, L. Rosasco, J. Mutch, A. Tacchetti, and T. Poggio, "Unsupervised learning of invariant representations," *Theor. Comput. Sci.*, vol. 633, pp. 112–121, Jun. 2016.

[15] I. Sharafaldin, A. H. Lashkari, S. Hakak, and A. A. Ghorbani, "Developing realistic distributed denial of service (DDoS) attack dataset and taxonomy," in *Proc. Int. Carnahan Conf. Secur. Technol. (ICCST)*, Oct. 2019, pp. 1–8.

[16] O. I. Abiodun, A. Jantan, A. E. Omolara, K. V. Dada, N. A. Mohamed, and H. Arshad, "State-of-the-art in artificial neural network applications: A survey," *Heliyon*, vol. 4, no. 11, Nov. 2018, Art. no. e00938.

[17] A. Tealab, "Time series forecasting using artificial neural networks methodologies: A systematic review," *Future Comput. Informat. J.*, vol. 3, no. 2, pp. 334–340, Dec. 2018.

[18] J. L. Elman, "Finding structure in time," *Cognit. Sci.*, vol. 14, no. 2, pp. 179–211, Mar. 1990.

[19] A. Sherstinsky, "Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network," *Phys. D, Nonlinear Phenomena*, vol. 404, Mar. 2020, Art. no. 132306.

[20] V. K. Navya, J. Adithi, D. Rudrawal, H. Tailor, and N. James, "Intrusion detection system using deep neural networks (DNN)," in *Proc. Int. Conf. Advancements Electr., Electron., Commun., Comput. Autom. (ICAECA)*, Oct. 2021, pp. 1–6.

[21] S. Romanosky, "Examining the costs and causes of cyber incidents," *J. Cybersecur.*, vol. 2, no. 2, pp. 121–135, 2016.

[22] B. Bhushan, G. Sahoo, and A. K. Rai, "Man-in-the-middle attack in wireless and computer networking—A review," in *Proc. 3rd Int. Conf. Adv. Comput., Commun. Autom. (ICACCA) (Fall)*, Sep. 2017, pp. 1–6.

[23] J. Clarke, *SQL Injection Attacks Defense*. Amsterdam, The Netherlands: Elsevier, 2009.

[24] P. Tang, W. Qiu, Z. Huang, H. Lian, and G. Liu, "Detection of SQL injection based on artificial neural network," *Knowl.-Based Syst.*, vol. 190, Feb. 2020, Art. no. 105528.

[25] M. Raza, M. Iqbal, M. Sharif, and W. Haider, "A survey of password attacks and comparative analysis on methods for secure authentication," *World Appl. Sci. J.*, vol. 19, no. 4, pp. 439–444, 2012.

[26] L. Bošnjak, J. Sreš, and B. Brumen, "Brute-force and dictionary attack on hashed real-world passwords," in *Proc. 41st Int. Conv. Inf. Commun. Technol., Electron. Microelectron. (MIPRO)*, May 2018, pp. 1161–1166.

[27] H. Hasbullah and I. A. Soomro, "Denial of service (DOS) attack and its possible solutions in VANET," *Int. J. Electron. Commun. Eng.*, vol. 4, no. 5, pp. 813–817, 2010.

[28] S. Mahjabin, "Implementation of DoS and DDoS attacks on cloud servers," *Periodicals Eng. Natural Sci.*, vol. 6, no. 2, pp. 148–158, 2018.

[29] K. Verma, H. Hasbullah, and A. Kumar, "An efficient defense method against UDP spoofed flooding traffic of denial of service (DoS) attacks in VANET," in *Proc. 3rd IEEE Int. Advance Comput. Conf. (IACC)*, Feb. 2013, pp. 550–555.

[30] M. M. Najafabadi, T. M. Khoshgoftaar, A. Napolitano, and C. Wheelus, "Rudy attack: Detection at the network level and its important features," in *Proc. 29th Int. Flairs Conf.*, 2016, pp. 282–287.

[31] K. N. Mallikarjunan, K. Muthupriya, and S. M. Shalinie, "A survey of distributed denial of service attack," in *Proc. 10th Int. Conf. Intell. Syst. Control (ISCO)*, Jan. 2016, pp. 1–6.

[32] K. Sonar and H. Upadhyay, "A survey: DDos attack on Internet of Things," *Int. J. Eng. Res. Develop.*, vol. 10, no. 11, pp. 58–63, 2014.

[33] B. Sieklik, R. Macfarlane, and W. J. Buchanan, "Evaluation of TFTP DDoS amplification attack," *Comput. Secur.*, vol. 57, pp. 67–92, Mar. 2016.

[34] T. Shorey, D. Subbaiah, A. Goyal, A. Sakxena, and A. K. Mishra, "Performance comparison and analysis of slowloris, GoldenEye and xerxes DDoS attack tools," in *Proc. Int. Conf. Adv. Comput., Commun. Informat. (ICACCI)*, Sep. 2018, pp. 318–322.

[35] G. Singh and B. Singh, "Simple service discovery protocol based distributed reflective denial of service attack," *Int. J. Recent Trends Eng. Res.*, vol. 3, no. 12, pp. 143–150, 2017.

[36] S.-J. Choi and J. Kwak, "A study on reduction of DDoS amplification attacks in the UDP-based CLDAP protocol," in *Proc. 4th Int. Conf. Comput. Appl. Inf. Process. Technol. (CAIPT)*, Aug. 2017, pp. 1–4.

[37] F. Hussain, S. G. Abbas, M. Husnain, U. U. Fayyaz, F. Shahzad, and G. A. Shah, "IoT DoS and DDoS attack detection using ResNet," in *Proc. IEEE 23rd Int. Multitopic Conf. (INMIC)*, Nov. 2020, pp. 1–6.

[38] M. Bogdanoski, T. Shuminoski, and A. Risteski, "Analysis of the SYN flood DoS attack," *Int. J. Comput. Netw. Inf. Secur.*, vol. 5, pp. 1–11, Jun. 2013.

[39] M. Najafimehr, S. Zarifzadeh, and S. Mostafavi, "A hybrid machine learning approach for detecting unprecedented DDoS attacks," *J. Supercomput.*, vol. 78, no. 6, pp. 8106–8136, Apr. 2022.

[40] T. R. Sree and S. M. S. Bhanu, "Detection of HTTP flooding attacks in cloud using fuzzy bat clustering," *Neural Comput. Appl.*, vol. 32, no. 13, pp. 9603–9619, Jul. 2020.

[41] S. Mansfield-Devine, "The growth and evolution of DDoS," *Netw. Secur.*, vol. 2015, no. 10, pp. 13–20, Oct. 2015.

[42] M. L. Mavrovouniotis and S. Chang, "Hierarchical neural networks," *Comput. Chem. Eng.*, vol. 16, no. 4, pp. 347–369, 1992.

[43] C. Zhang, J. Jiang, and M. Kamel, "Intrusion detection using hierarchical neural networks," *Pattern Recognit. Lett.*, vol. 26, no. 6, pp. 779–791, May 2005.

[44] G. Bartfai, "Hierarchical clustering with ART neural networks," in *Proc. IEEE Int. Conf. Neural Netw. (ICNN)*, vol. 2, Jun. 1994, pp. 940–944.

[45] D. Dua and C. Graff, "UCI machine learning repository," School Inf. Comput. Sci., Univ. California, Irvine, CA, USA, 2019. [Online]. Available: http://archive.ics.uci.edu/ml

[46] S. Mohammadi, H. Mirvaziri, M. Ghazizadeh-Ahsaee, and H. Karimipour, "Cyber intrusion detection by combined feature selection algorithm," *J. Inf. Secur. Appl.*, vol. 44, pp. 80–88, Feb. 2019.

[47] A. Hadiks, Y. Chen, F. Li, and B. Liu, "A study of stealthy denial-of-service attacks in Wi-Fi direct device-to-device networks," in *Proc. IEEE 11th Consumer Commun. Netw. Conf. (CCNC)*, Jan. 2014, pp. 507–508.

[48] H. S. Chae, B. O. Jo, S. H. Choi, and T. K. Park, "Feature selection for intrusion detection using NSL-KDD," *Recent Adv. Comput. Sci.*, vol. 20132, pp. 184–187, Nov. 2013.

[49] A. Mahfouz, A. Abuhussein, D. Venugopal, and S. Shiva, "Ensemble classifiers for network intrusion detection using a novel network attack dataset," *Future Internet*, vol. 12, no. 11, p. 180, Oct. 2020.

[50] B. Hussain, Q. Du, B. Sun, and Z. Han, "Deep learning-based DDoS-attack detection for cyber–physical system over 5G network," *IEEE Trans. Ind. Informat.*, vol. 17, no. 2, pp. 860–870, Feb. 2021.

[51] Y. Al-Nashif, A. A. Kumar, S. Hariri, Y. Luo, F. Szidarovsky, and G. Qu, "Multi-level intrusion detection system (ML-IDS)," in *Proc. Int. Conf. Autonomic Comput.*, Jun. 2008, pp. 131–140.

[52] J. Cheng, C. Zhang, X. Tang, V. S. Sheng, Z. Dong, and J. Li, "Adaptive DDoS attack detection method based on multiple-kernel learning," *Secur. Commun. Netw.*, vol. 2018, pp. 1–19, Oct. 2018.

[53] P. Tam, R. Corrado, C. Eang, and S. Kim, "Applicability of deep reinforcement learning for efficient federated learning in massive IoT communications," *Appl. Sci.*, vol. 13, no. 5, p. 3083, Feb. 2023.

[54] M. Alqahtani, "Denial of service attack on D2D communication For 4G," *Int. J. Elect., Electron. Data Commun.*, vol. 5, no. 1, pp. 38–40, Jan. 2017.

[55] D. Barik, J. Sanyal, and T. Samanta, "Prevention of denial-of-service attacks in 5G D2D wireless communication networks," in *Proc. IEEE 3rd 5G World Forum (5GWF)*, Sep. 2020, pp. 239–244.

[56] S. V. J. Rani, I. Ioannou, P. Nagaradjane, C. Christophorou, V. Vassiliou, S. Charan, S. Prakash, N. Parekh, and A. Pitsillides, "Detection of DDoS attacks in D2D communications using machine learning approach," *Comput. Commun.*, vol. 198, pp. 32–51, Jan. 2023.

[57] M. Kjaerland, "A taxonomy and comparison of computer security incidents from the commercial and government sectors," *Comput. Secur.*, vol. 25, no. 7, pp. 522–538, Oct. 2006.

[58] M. Wang and Z. Yan, "Security in D2D communications: A review," in *Proc. IEEE Trustcom/BigDataSE/ISPA*, vol. 1, Aug. 2015, pp. 1199–1204.

[59] I. Ioannou, C. Christophorou, V. Vassiliou, and A. Pitsillides, "A novel distributed AI framework with ML for D2D communication in 5G/6G networks," *Comput. Netw.*, vol. 211, Jul. 2022, Art. no. 108987.

[60] W. L. Pritchett and D. De Smet, *Kali Linux Cookbook*. Birmingham, U.K.: Packt Publishing Ltd, 2013.

[61] N. H. Tanner, "Metasploit," in *Wiley Data and Cybersecurity*. Hoboken, NJ, USA: Wiley, 2019.

[62] M. Satheesh, B. J. D'mello, and J. Krol, *Web Development With MongoDB and NodeJs*. Birmingham, U.K.: Packt Publishing Ltd, 2015.

[63] Rapid7, Boston, MA, USA. *Download Metasploitable Intentionally Vulnerable Machine | Rapid7*. Accessed: Aug. 31, 2023. [Online]. Available: https://www.rapid7.com/products/metasploit/download/

[64] University of New Brunswick Canadian Institute for Cybersecurity, Fredericton, NB, Canada. *Applications | Research | Canadian Institute for Cybersecurity | UNB CICFlowMeter (Formerly ISCXFlowMeter)*. Accessed: Aug. 31, 2023. [Online]. Available: https://www.unb.ca/cic/research/applications.html

[65] A. Chartuni and J. Márquez, "Multi-classifier of DDoS attacks in computer networks built on neural networks," *Appl. Sci.*, vol. 11, no. 22, p. 10609, Nov. 2021.

[66] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique," *J. Artif. Intell. Res.*, vol. 16, pp. 321–357, Jun. 2002.

[67] University of New Brunswick Canadian Institute for Cybersecurity, Fredericton, NB, Canada. *DDoS 2019 | Datasets | Research | Canadian Institute for Cybersecurity | UNB*. Accessed: Aug. 31, 2023. [Online]. Available: https://www.unb.ca/cic/datasets/ddos-2019.html

[68] H. K. Gajera, M. A. Zaveri, and D. R. Nayak, "Patch-based local deep feature extraction for automated skin cancer classification," *Int. J. Imag. Syst. Technol.*, vol. 32, no. 5, pp. 1774–1788, Sep. 2022.

[69] N. Hoque, D. K. Bhattacharyya, and J. K. Kalita, "MIFS-ND: A mutual information-based feature selection method," *Expert Syst. Appl.*, vol. 41, no. 14, pp. 6371–6385, Oct. 2014.

[70] M. Brunato and R. Battiti, "X-MIFS: Exact mutual information for feature selection," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2016, pp. 3469–3476.

[71] A. F. Agarap, "Deep learning using rectified linear units (ReLU)," 2018, *arXiv:1803.08375*.

[72] V. Martinek, "Cross-entropy for classification," Towards Data Sci., Toronto, ON, Canada, Jun. 2020. Accessed: Aug. 31, 2023. [Online]. Available: https://towardsdatascience.com/cross-entropy-for-classification-d98e7f974451

[73] M. Irenee, X. Hei, Y. Wang, W. Ji, and X. Jiang, "Network flow analytics: Multi-class classification of DDoS attacks based on OKNN," in *Proc. Int. Conf. Netw. Netw. Appl. (NaNA)*, Dec. 2020, pp. 271–276.

[74] F. Meghdouri, M. Bachl, and T. Zseby, "EagerNet: Early predictions of neural networks for computationally efficient intrusion detection," in *Proc. 4th Cyber Secur. Netw. Conf. (CSNet)*, Oct. 2020, pp. 1–7.

[75] N. B. Harikrishnan, "Confusion matrix, accuracy, precision, recall, F1 score," Anal. Vidhya, Gurgaon, India, Oct. 2019.

[76] R. A. Bradley and M. E. Terry, "Rank analysis of incomplete block designs: I. The method of paired comparisons," *Biometrika*, vol. 39, nos. 3–4, pp. 324–345, 1952.

[77] T. Fawcett, "An introduction to ROC analysis," *Pattern Recognit. Lett.*, vol. 27, no. 8, pp. 861–874, Jun. 2006.

[78] I. Chivers and J. Sleightholme, "An introduction to algorithms and the big O notation," in *Introduction to Programming With Fortran*. Cham, Switzerland: Springer, 2015, pp. 359–364. [Online]. Available: https://link.springer.com/book/10.1007/978-3-319-17701-4

[79] K. Kowsari, R. Sali, L. Ehsan, W. Adorno, A. Ali, S. Moore, B. Amadi, P. Kelly, S. Syed, and D. Brown, "HMIC: Hierarchical medical image classification, a deep learning approach," *Information*, vol. 11, no. 6, pp. 1–19, 2020.

[80] M. S. Im and V. R. Dasari, "Computational complexity reduction of deep neural networks," Mar. 2022, *arXiv:2207.14620*.

[81] *Computational Complexity Of Neural Networks*. Accessed: Nov. 23, 2022. [Online]. Available: https://lunalux.io/computational-complexity-of-neural-networks/

[82] P. Vadher, "Snort IDPS using raspberry Pi 4," *Int. J. Eng. Res.*, vol. V9, no. 7, pp. 151–154, Jul. 2020.

S. V. JANSI RANI received the B.E. degree (Hons.) in CSE from Bharathidasan University, the M.Tech. degree in advanced computing from SASTRA University, and the Ph.D. degree from Anna University, Chennai. She is an Associate Professor with the Department of Computer Science, and has more than 14 years of teaching experience. She has authored a book on data mining and data warehousing. She has a strong striving passion for her profession. She is a Co-Investigator of a faculty internally funded project titled "Predictive Policing Using Social Network Analysis," funded by the SSN Trust. She has published many papers in international journals and conferences. Her research interests include computer network protocols, machine learning, data analytics, and social network analysis. She has organized many workshops, such as Multicore Computing and Programming and Computational Thinking.

IACOVOS I. IOANNOU received the Associate degree in computer science from the Cyprus College, in 2001, the B.Sc. degree in computer science from the University of Cyprus, in 2006, the M.Sc. degree (Hons.) in computer and network security from the Open University of Cyprus, in 2017, and the Ph.D. degree from the University of Cyprus, in 2021, focusing on telecommunications with AI/ML.

He is a Researcher with the Networks Research Laboratory, University of Cyprus. He is also a Junior Researcher with the Smart Networked Systems Research Group, RISE Center. He is a highly skilled developer with 20 years of hands-on working experience in IT industry, ranging in the spectrum of IT systems from analysis, development, installation, and management. He was with the Phileleftheros Publishing Group, as an IT Administrator and a Programmer, for seven years. He was also with the Cyprus Stock Exchange, as an IT and Programmer, for seven years, and with Primetel, as an IT and Software Engineer with the Services Department, for six years. He has vast experience in cellular network infrastructure and all modern development platforms and languages. He has a certificate in ICONIX/SCRUM and has a CCNET certificate from CISCO. His research interests include mobile and wireless communications, next-generation networks (5G), and device-to-device (D2D) communications using artificial intelligence techniques.

PRABAGARANE NAGARADJANE received the M.Tech. and Ph.D. degrees in ECE from the Pondicherry Engineering College, Pondicherry (Central) University. Prior to this, he was with HCL Info Systems, for two years. He is an Associate Professor with the Department of Electronics and Communication Engineering, SSN Institutions. He has authored or coauthored more than 50 technical articles. His research interests include various aspects of wireless communications, especially with respect to signal processing for wireless and broadband communications. He is one of the founding members of the Wireless Communications, Signal Processing and Networking (WiSPNET) International Conference, technically co-sponsored by IEEE. He has served as one of the organizing chairs for 2016 WiSPNET, 2017 WiSPNET, 2018 WiSPNET, 2020 WiSPNET, and 2021 WiSPNET; and the Co-Chair for the WiSPNET 2019 Conference. He is serving as one of the organizing chairs for the WiSPNET 2022 Conference. He has guest-edited two special issues under the topic of signal processing for 5G in the Elsevier *Computers and Electrical Engineering*. He also served on the editorial board for the *Physical Communication* as an Area Editor. He serves as an Associate Editor for the *IET Communications*.

CHRISTOPHOROS CHRISTOPHOROU received the B.Sc. degree in computer science, the M.Sc. degree in advanced computer technologies, and the Ph.D. degree in mobile/wireless networks from the University of Cyprus, in 2002, 2005, and 2011, respectively.

He has published over 35 articles in journals, scientific conferences, and book chapters. Since 2004, he has been involving initially as a researcher and later as a project manager in different local (RPF-funded) and EU-funded research projects in the domain of AAL and AgeingWell, such as FRAIL, SUCCESS, MEMENTO, GrowMeUp, miraculous-life, socialrobot, and co-living; and in the mobile and wireless networks domain, such as C-CAST, C-MOBILE, MOTIVE, and BBONE. His main research interests and expertise are mostly concentrated in the telecommunications and networking research, database management systems, social collaborative care networks, and information and communication technology (ICT) personalized solutions (for various sectors including the e-health domain).

VASOS VASSILIOU (Senior Member, IEEE) is an Associate Professor with the Computer Science Department, University of Cyprus, where he is also the Co-Director of the Networks Research Laboratory (NetRL), founded by Prof. A. Pitsillides. He has been a Group Leader of the Smart Networked Systems Research Group, RISE Center of Excellence on Interactive Media, Smart Systems and Emerging Technologies, Nicosia, Cyprus, since November 2017. He was appointed by the Senate of the University of Cyprus to the Board of Directors of CYNET, the National Research and Educational Network, where he has been the Chair, since November 2016. He has more than 75 publications in academic journals, books, and international conferences, with more than 1300 citations. His research interests include protocol design and performance aspects of networks (fixed, mobile, and wireless), in particular mobility management, QoS adaptation and control, resource allocation techniques, wireless sensor networks, and the Internet of Things.

He is a Senior Member of ACM. He is a Treasurer of the IEEE Cyprus Section, the Chair of the IEEE ComSoc Cyprus Chapter, a part of the IBM Academic Initiative, and an Academic Advocate of the ISACA Cyprus Chapter. He has been a member of the TPC of about 100 international conferences in his field and acted as the chair of various committees in more than 15 conference organizations. He is also in the editorial boards of the *Telecommunication Systems* (Springer) and the *Computer Networks* (Elsevier).

HARSHITAA YARRAMSETTI received the bachelor's degree in information technology from the Sri Sivasubramaniya Nadar College of Engineering, Anna University, Chennai, India. She is currently pursuing the master's degree in computer science with North Carolina State University, USA. She is committed to working on projects that use machine learning and deep learning to tackle problem statements and simultaneously explore the field effectively.

SAI SHRIDHAR received the bachelor's degree in computer science engineering from the Sri Sivasubramaniya Nadar College of Engineering, Anna University, Chennai, India. He is currently pursuing the master's degree with the University of Illinois at Chicago, USA. He is a dedicated researcher in the field of computer science. His research focuses on machine learning, neural networks, and natural language processing.

L. MUKUND BALAJI received the bachelor's degree in computer science from the Sri Sivasubramaniya Nadar College of Engineering, Anna University, Chennai, India. He is currently pursuing the master's degree in computer science with North Carolina State University, USA. His research interests include using machine learning models and data analysis to solve real-world problems.

ANDREAS PITSILLIDES (Life Senior Member, IEEE) is a Professor with the Department of Computer Science, University of Cyprus. He was the Co-Director of the Networks Research Laboratory (NetRL, http://www.NetRL.cs.ucy.ac.cy), and a Visiting Professor with the Department of Electrical and Electronic Engineering Science, University of Johannesburg, South Africa, from 2021 to 2024. He was appointed as a Visiting Professor with the School of Electrical and Information engineering, University of the Witwatersrand (Wits), from 2017 to 2020, and the Department of Electrical and Electronic Engineering Science, University of Johannesburg, from 2014 to 2017. He has participated as a principal or a co-principal investigator in over 40 European Commission and locally funded research projects with over 6.7 million Euro. He has published over 350 refereed articles in flagship journals (e.g., IEEE, Elsevier, IFAC, and Springer), international conferences, and book chapters; and coauthored two books (one edited). His broad research interests include communication networks, software defined metamaterials (hypersurfaces and reconfigurable intelligent surfaces), nanonetworks, the Internet- and Web- of Things, smart systems (e.g., smart grid), smart spaces (e.g., home and city), and e-health. He has a particular interest in adapting tools from various fields of applied mathematics, such as adaptive non-linear control theory, computational intelligence, game theory, and complex systems and nature inspired techniques, to solve problems in communication networks.

He received several awards, including best paper, presented keynotes, invited lectures at major research organizations, short courses at international conferences, and short courses to industry.

• • •